RM03

FUNCTIONAL TEST PART 2

MD-11-DZRMD-A

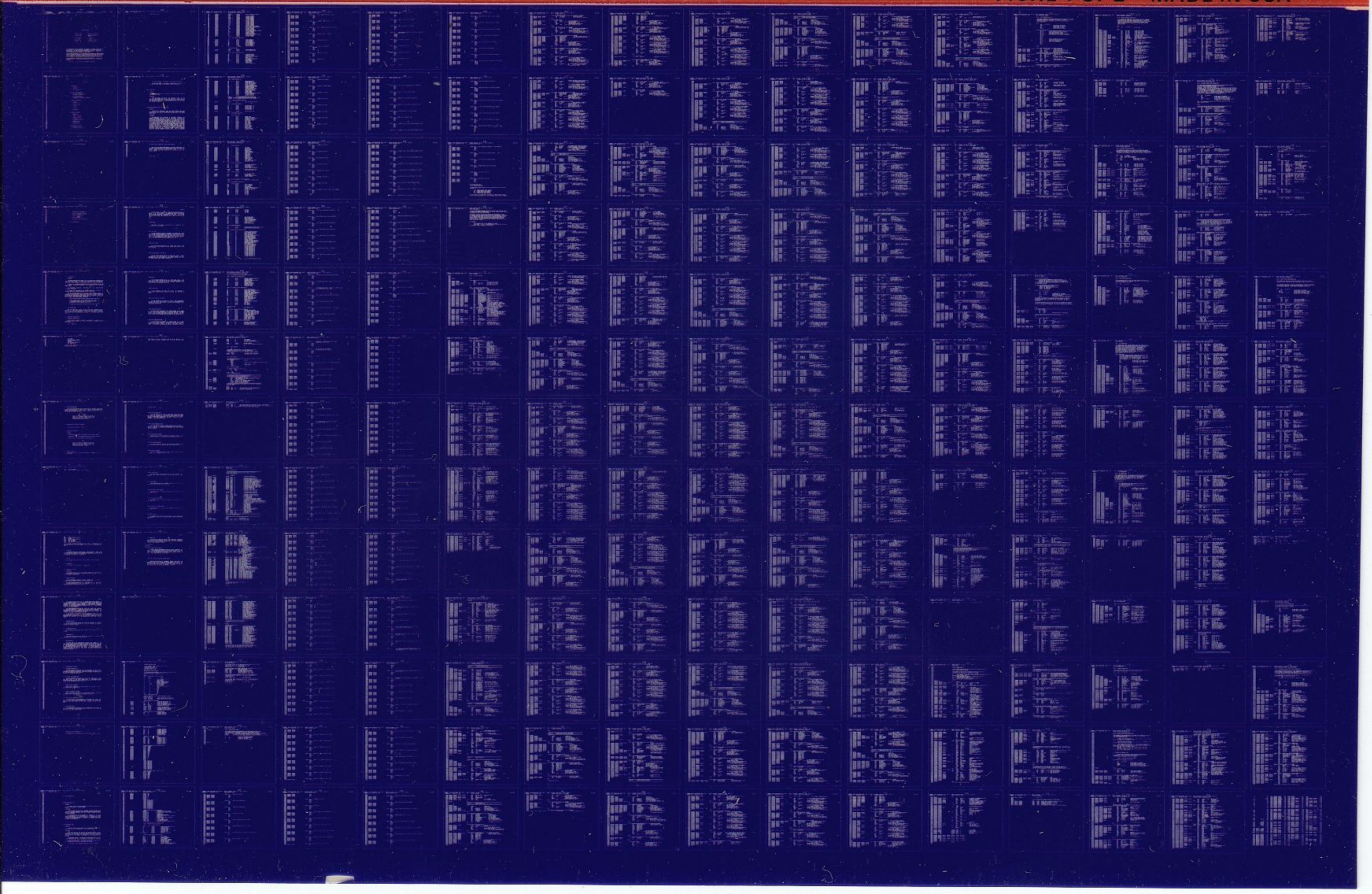
EP-DZRMD-A-DL-A

OCT 1977

digital

FICHE 1 OF 2

MADE IN USA



RM03

FUNCTIONAL TEST PART 2

MD-11-DZRMD-A

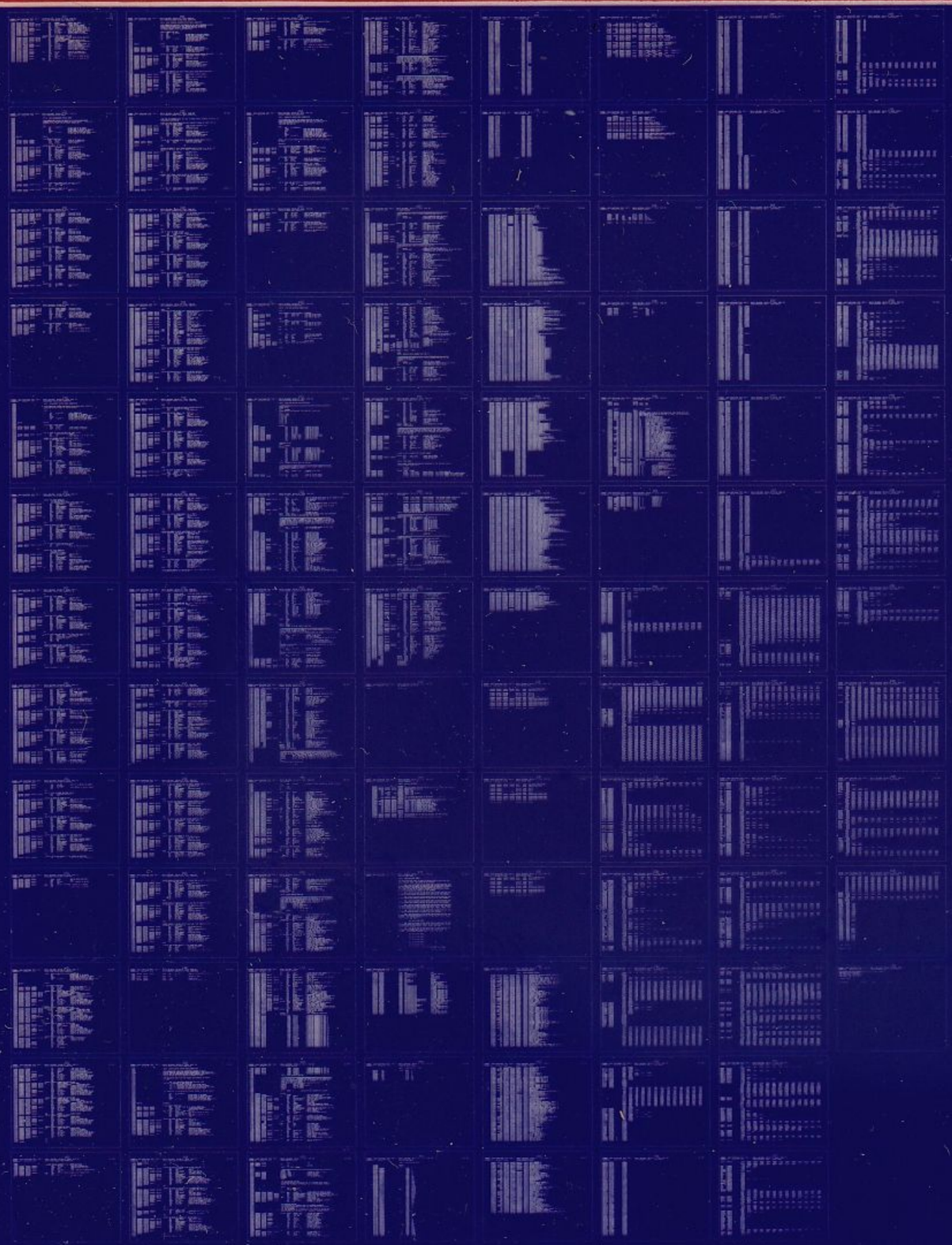
EP-DZRMD-A-DL-A

FICHE 2 OF 2

OCT 1977

digital

MADE IN USA



```
                        .REM    \
```

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZRMD-A-D
PRODUCT NAME:	RM03 FUNCTIONAL TEST, PART 2
DATE CREATED:	1 AUGUST 77
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	DOUG RIIKONEN

```
                        .PAGE
```

CONTENTS

109 2. DUAL PORT CONFIGURATIONS

3. MEMORY PARITY HARDWARE

4. MEMORY MANAGEMENT HARDWARE

5. ACT,APT COMPATIBILITY

6. XXDP COMPATIBILITY

7. OPERATING SYSTEM COMPATIBILITY

6. TEST DESCRIPTION

1.0 INTRODUCTION

1.1 ABSTRACT

THE RM03 SUBSYSTEM FUNCTIONAL TEST IS A STAND ALONE PROGRAM WHICH USES FUNCTIONAL MEANS TO VERIFY THE OPERABILITY OF THE RM03 DISK SUBSYSTEM. IN PARTICULAR, THE PROGRAM SERVES THE FOLLOWING PURPOSES:

TO EXPLICITLY ESTABLISH CONFIDENCE IN THE BASIC OPERATIONS OF THE DISK DRIVE, INCLUDING MECHANICAL POSITIONING AND DATA TRANSFER OPERATIONS;

TO IMPLICITLY ESTABLISH CONFIDENCE IN THE DRIVE/ADAPTER ELECTRICAL INTERFACE;

TO VERIFY THE FUNCTIONALITY OF THE RM03 SUBSYSTEM, INCLUDING THE MASSBUS CONTROLLER, MASSBUS ADAPTER AND THE DISK DRIVE.

THE TEST IS COMPRISED OF 3 PARTS, WHICH WOULD NORMALLY BE RUN IN SEQUENCE, STARTING WITH PART 1. BRIEFLY, PART 1 TESTS HOUSEKEEPING AND MECHANICAL POSITIONING OPERATIONS; PART 2 TESTS WRITE, READ AND WRITE CHECK OPERATIONS USING HEADER AND DATA; PART 3 TESTS WRITE, READ AND WRITE CHECK OPERATIONS USING DATA.

```
**********************************************************
*                                                        *
* NOTE: PARTS 2 AND 3 OF THE FUNCTIONAL TEST LEAVE       *
* 2 HEADER ERRORS ON THE MEDIA. THE PACK SHOULD BE       *
* REFORMATTED AFTER RUNNING THESE TESTS.                 *
*                                                        *
**********************************************************
```

1.2 UNIT UNDER TEST

THE UNIT UNDER TEST (UUT) IS THE RM03 DISK SUBSYS 1 WHICH CONSISTS OF THE RHXX MASSBUS CONTROLLER, THE RM03 MASSBUS ADAPTER, AND THE STORAGE MODULE DISK DRIVE. NOTE THAT A DISK PACK IS REQUIRED FOR TESTING AND IS CONSIDERED AN INTEGRAL OF THE STORAGE MODULE DISK DRIVE.

2.0 OPERATING REQUIREMENTS

2.1 HARDWARE REQUIREMENTS

THE FOLLOWING MINIMUM HARDWARE CONFIGURATION, ASSUMED TO BE OPERATIONAL, IS REQUIRED TO LOAD AND EXECUTE THE RM03 SUBSYSTEM FUNCTIONAL TEST:

PDP-11 PROCESSOR

```
                                        16K MEMORY
                                        KW11-L OR KW11-P CLOCK
                                        PROGRAM LOADING DEVICE
                                        TERMINAL
                                        RH11 OR RH70 CONTROLLER
                                        UNIT UNDER TEST,

                                    WHERE THE UNIT UNDER TEST CONSISTS OF ONE TO EIGHT RM03 ADAPTERS, DISK
                                    DRIVES AND DISK PACKS.
```

2.2 MEDIA REQUIREMENTS

EACH UNIT BEING TESTED MUST BE LOADED WITH A SCRATCH DISK PACK BEFORE TESTING BEGINS ON THAT UNIT. THE PACK MAY BE FORMATTED OR UNFORMATTED, BUT MUST NOT CONTAIN NEEDED INFORMATION BECAUSE THE PACK WILL BE WRITTEN DURING THE TEST.

NOTE

WHEN THE PROGRAM IS LOADED VIA THE RM03, DRIVE 0 IS NOT TESTED UNLESS THE OPERATOR SPECIFICALLY ENTERS THE DRIVE NUMBER DURING CONSOLE DIALOGUE.

2.3 PREREQUISITE DIAGNOSTIC PROGRAMS

RM03 DISKLESS DIAGNOSTIC, DZRMJ-A

3.0 OPERATING PROCEDURE

3.1 LOADING

THE PROGRAM MAY BE LOADED BY EITHER OF THE FOLLOWING MEDIA:

. PAPER TAPE, USING THE STANDARD PAPER TAPE LOADING PROCEDURE;

.XXDP MEDIA, USING THE APPROPRIATE LOADING DEVICE.

NOTE

WHEN THE PROGRAM IS LOADED VIA THE RM03, DRIVE 0 IS NOT TESTED UNLESS THE OPERATOR SPECIFICALLY ENTERS THE DRIVE NUMBER DURING CONSOLE DIALOGUE.

3.2 SWITCH OPTIONS

THE FOLLOWING SWITCH OPTIONS ARE PROVIDED TO ENHANCE THE UTILITY OF THE PROGRAM.

```
   247                              SW15    HALT ON ERROR
   248                              SW14    LOOP ON TEST (CURRENTLY BEING EXECUTED)
```

SW13 INHIBIT ERROR TYPEOUTS
SW12 UNUSED
SW11 INHIBIT TEST ITERATIONS
SW10 BELL ON ERROR
SW09 LOOP ON ERROR
SW08 LOOP ON TEST IN SW07-00

THE LOW ORDER 8 SWITCHES (SW07-SW00), ARE USED IN CONJUNCTION WITH SW08 TO SPECIFY THE OCTAL NUMBER OF THE TEST WHICH THE PROGRAM WILL LOOP ON.

3.3 STARTING

THE PROGRAM STARTS AT LOCATION 200, WHICH PROVIDES WORST CASE TEST CONDITIONS IF RUNNING IN AN AUTOMATIC ENVIRONMENT. IF RUNNING IN A STAND-ALONE ENVIRONMENT, THE PROGRAM USES CONSOLE DIALOGUE TO ALLOW THE OPERATOR TO CONTROL TEST CONDITIONS.

3.4 HALTING

THE PROGRAM CAN BE HALTED BY TYPING CONTROL C FROM THE CONSOLE.

3.5 RESTARTING

THE PROGRAM CAN BE RESTARTED AT ADDRESS 200.

4.0 OPERATOR INTERFACE

4.1 PROGRAM ID

THE PROGRAM TYPES ITS TITLE AND MAINDEC NUMBER THE FIRST TIME IT IS STARTED AFTER BEING LOADED. PROGRAM IDENTIFICATION DOES NOT OCCUR IF THE PROGRAM IS RESTARTED.

4.2 CONSOLE DIALOGUE

WHEN THE PROGRAM IS RUNNING IN STAND ALONE MODE, IT ENTERS A CONSOLE DIALOGUE SEQNENCE AFTER TYPING THE PROGRAM I.D..

THE FIRST QUESTION TYPED OUT IS: "TYPE HELP TEXT (Y OR N)??". IF THE OPERATOR RESPONDS WITH A Y, THE PROGRAM WILL TYPE A BRIEF HELP MESSAGE WHCIH WILL LIST SWITCH OPTIONS, ETC.

THE SECOND QUESTION TYPED OUT IS, "CHANGE RM03 UNIBUS ADDRESS OR VECTOR ADDRESS (Y OR N)??". IF THE UNIBUS ADDRESS OF THE RM03 IS NON STANDARD, THE OPERATOR SHOULD RESPOND Y, THEN ANSWER SUBSEQUENT QUESTIONS TO SPECIFY THE UNIBUS ADDRESS, VECTOR ADDRESS AND INTERRUPT PRIORITY. IF THE OPERATOR DOES NOT RESPOND WITH A Y, THE PROGRAM SKIPS TO THE THIRD QUESTION.

THE THIRD QUESTION TYPED OUT IS, "TYPE (A) TO TEST ALL DEVICES, OR TYPE DEVICE NUMBER(S). TERMINATE INPUT WITH CARRIAGE RETURN". IF THE OPERATOR TYPES A, ALL POSSIBLE DEVICES ARE TESTED. OTHERWISE, THE OEPRATOR CAN TYPE THE NUMBER(S) OF THE DEVICE(S) HE WANTS TESTED, AND TERMINATE HIS INPUT WITH A CARRIAGE RETURN.

IF THE PROGRAM IS RESTARTED, THE FIRST QUESTION TYPED IS, "USE SAME DEVICES (Y OR N)??". IF THE OPERATOR TYPES Y, THE TEST IS RESTARTED USING THE SAME DEVICES AS THE LAST TIME, OTHERWISE, THE TEST RESTARTS THE DIALOGUE AS IF THE PROGRAM WERE STARTED FOR THE FIRST TIME.

4.3 PROGRESS REPORTS

AN END OF PASS REPORT OCCURS EACH TIME THE PROGRAM IS EXECUTED FOR ALL DEVICES IN THE TEST QUE. THE END OF PASS REPORT INCLUDES A MESSAGE AND AN ERROR SUMMARY.

4.4 PERFORMANCE REPORT

NO PERFORMANCE REPORTS ARE GIVEN DURING THE EXECUTION OF THE PROGRAM.

4.5 PROGRAM HALTS

THERE ARE NO SCHEDULED HALTS DURING THE EXECUTION OF THE PROGRAM. PROCESSOR HALTS ARE DUE TO THE TRAP CATCHER.

4.6 ERROR REPORTS

THE FIRST LINE OF THE ERROR REPORT CONTAINS THE NUMBER OF THE UNIT BEING TESTED, THE TEST NUMBER, THE ERROR NUMBER AND THE VALUE OF THE PROGRAM COUNTER WHERE THE ERROR WAS CALLED. THIS LINE IS FOLLOWED BY THE ERROR MESSAGE: ONE OR MORE LINES OF TEXT WHICH GIVE A BRIEF, YET COMPREHENSIVE DESCRIPTION OF THE ERROR. THE ERROR MESSAGE IS NORMALLY FOLLOWED BY ONE OR MORE PAIRS OF LINES CONTAINING DATA HEADERS AND DATA PERTININENT TO THE ERROR, INCLUDING EXPECTED AND ACTUAL TEST RESULTS.

5.0 ENVIRONMENTAL SUPPORT

5.1 PROCESSOR COMPATIBILITY

THE RM03 SUBSYSTEM FUNCTIONAL TEST IS EXECUTABLE ON ANY PDP-11 PROCESSOR, PROVIDING PREVIOUSLY MENTIONED HARDWARE REQUIREMENTS ARE MET, AND PROVIDING THAT DATA THROUGHPUT ON THE SYSTEM IS SUFFICIENT TO SUSTAIN DATA TRANSFER OPERATIONS.

5.2 DUAL PORT CONFIGURATIONS

THE RM03 SUBSYSTEM FUNCTIONAL TEST DOES NOT SPECIFICALLY TEST DUAL PORT LOGIC IN THE RM03 ADAPTER BUT IS EXECUTABLE ON RM03 SUBSYSTEMS HAVING THE DUAL PORT OPTION PROVIDING THE DUAL PORT SWITCH IS SET TO THE APPROPRIATE PORT (A OR B).

5.3 MEMORY PARITY HARDWARE

MEMORY PARITY HARDWARE IS NOT USED DURING THE EXECUTION OF THE RM03 SUSBYSTEM FUNCTIONAL TEST.

5.4 MEMORY MANAGEMENT HARDWARE

MEMORY MANAGEMENT HARDWARE IS NOT USED DURING THE RM03 SUSBYSTEM FUNCTIONAL TEST. CAPABILITIES OF THE MASSBUS CONTROLLER.

5.5 ACT11, APT11 COMPATIBILITY

THE RM03 SUSBYSTEM FUNCTIONAL TEST IS COMPATIBLE WITH ACT11 AND APT11 IN BOTH DUMP AND AUTOMATIC MODES. FURTHER, THE PROGRAM WILL EXECUTE A QUICK PASS DURING THE FIRST PASS IN SUPPORT OF QUICK VERIFY MODE.

5.6 XXDP COMPATIBILITY

THE RM03 SUSBYSTEM FUNCTIONAL TEST IS COMPATIBLE WITH XXDP IN DUMP AND CHAIN MODES, AND PROVIDES MEDIA PROTECTION IN THE CASE WHERE THE RM03 IS THE XXDP LOADING DEVICE.

5.7 OPERATING SYSTEM COMPATIBILITY

THE PROGRAM IS NOT COMPATIBLE WITH ANY SOFTWARE OPERATING SYSTEM.

6.0 TEST DESCRIPTION

CONTROLLER ACCESS TEST

PURPOSE:

TO VERIFY THAT THE UNIBUS ADDRESS OF THE RM03 SUBSYSTEM IS CORRECT, AS DEFINED AT LOCATION $BASE.

PROCEDURE:

THE TEST TRIES TO ACCESS ALL MASSBUS CONTROLLER REGISTERS USING THE $BASE ADDRESS. REGISTER CONTENTS ARE IGNORED DURING THE TEST, AND THE TEST FAILS IF A BUS TIMEOUT OCCURS FOR ANY REGISTER TRANSFER.

IF THE TEST FAILS AND THE PROGRAM IS RUNNING IN A STAND ALONE ENVIRONMENT, I.E., LOCATION 42 IS 0, THE PROGRAM WILL JUMP TO LOCATION 204 WHICH ALLOWS THE OPERATOR TO CHANGE THE $BASE ADDRESS VIA CONSOLE DIALOGUE. OTHERWISE, THE PROGRAM ESCAPES TO THE END OF PASS HANDLER.

DEVICE AVAILABLE TEST

PURPOSE:

TO VERIFY THAT THE UNIT UNDER TEST IS AVAILABLE FOR TESTING, AND IS NOT LOCKED OR PROGRAMMED TO THE ALTERNATE PORT.

PROCEDURE:

THIS TEST SELECTS THE DEVICE AND READS CONTROL STATUS REGISTERS 1 AND 2 TO VERIFY THAT THE SELECTED DEVICE IS AVAILABLE FOR TESTING, AS INDICATED BY DVA STATUS, BIT 11 OF RMCS1 AND NED STATUS, BIT 12 OF RMCS2. THE RESULTS OF THE TEST CAN VARY AS FOLLOWS:

.NONEXISTANT DEVICE - THE DEVICE IS NONEXISTENT OR IS LOCKED ON THE ALTERNATE PORT AND IS THEREFORE NOT AVAILABLE FOR TESTING;

.DEVICE NOT AVAILABLE - THE DEVICE EXISTS BUT IS SEIZED BY THE ALTERNATE PORT AND IS NOT AVAILABLE FOR TESTING;

.BUS TIMEOUT - THE MASSBUS CONTROLLER FAILED TO DETECT A

```
   474                                  NONEXISTENT DEVICE;
```

.DEVICE AVAILABLE - THE DEVICE IS AVAILABLE FOR TESTING.

THE PROGRAM JUMPS TO THE SUBPASS HANDLER WHICH SELECTS THE NEXT DRIVE TO BE TESTED IF THE DEVICE IS NOT AVAILABLE.

DRIVE TYPE TEST

PURPOSE:

TO VERIFY THAT THE UNIT UNDER TEST IS AN RM03 SINGLE PORT OR DUAL PORT SUBSYSTEM.

PROCEDURE:

THIS TEST READS THE DRIVE TYPE REGISTER, RMDT, OF THE SELECTED DEVICE AND VERIFIES THAT THE DEVICE IS A SINGLE PORT OR DUAL PORT RM03 SUBSYSTEM. IF THE SELECTED DEVICE IS NOT AN RM03, THE PROGRAM JUMPS TO THE SUBPASS HANDLER WHICH WILL SELECT THE NEXT DEVICE FOR TESTING.

WRITE/READ HEADER AND DATA (FORMAT) TESTS

PURPOSE:

TO TEST WRITE HEADER AND DATA AND READ HEADER AND DATA FUNCTIONALITY OF THE RM03 SUBSYSTEM USING A SET OF VARIABLES WHICH INCLUDE WORD COUNT, HEAD MOTION, HEAD SWITCHING AND ERROR CONDITIONS.

PROCEDURE:

ALTHOUGH EACH TEST EXERCISES A DIFFERENT VARIABLE, THE GENERAL PROCEDURE OF EACH TEST IS THE SAME. THE DRIVE IS INITIALIZED AND RECALIBRATED IF "PIP" OR "SKI" ARE ACTIVE SO THAT THERE ARE NO ERRORS WHEN A TEST BEGINS. FOLLOWING THAT, THE TEST PERFORMS ANY EXPLICIT SEEKS REQUIRED FOR THE CONDITIONS OF THE TEST. REGISTERS ARE PRESET AND THE WRITE HEADER AND DATA COMMAND IS EXECUTED. WHEN THE WRITE COMMAND IS COMPLETE, THE TEST STORES ALL SUBSYSTEM STATUS AND CHECKS FOR PRIMARY ERRORS WHICH PRECLUDE OTHER STATUS CHECKS. IF THERE ARE NO PRIMARY ERRORS, THE TEST VERIFIES THE RESULTS OF THE WRITE COMMAND AND THEN CHECKS FOR SECONDARY ERRORS. LOOP ADDRESSES ARE MODIFIED FOLLOWING THE SUCCESSFUL COMPLETION OF THE WRITE COMMAND IN ORDER TO SHORTEN EXECUTION TIMES AND ENHANCE SCOPING LOOPS. THEN THE PROGRAM EXECUTES THE READ HEADER AND DATA PORTION OF THE TEST, VERIFYING

```
                                THE SAME TYPE OF ERRORS AS IN THE WRITE COMMAND.


                                     NOTE THAT THE SECTOR USED DURING A TEST MAY DIFFER FROM  THE
                                PROGRAM  LISTING BECAUSE THE PROGRAM SUBSTITUTES A GOOD SECTOR IF
                                THE ONE SELECTED IS LISTED IN THE BAD BLOCK  TABLE  OF  THE  LAST
                                TRACK.  UNLESS  SPECIFIED OTHERWISE, ALL TESTS ARE IN 16 BIT (32
                                SECTOR) FORMAT.




                                FORMAT ZEROS - 18
```

THE TEST SEEKS TO CYLINDER 0, THEN WRITES HEADER AND DATA ON SECTOR 0 IN 18 BIT FORMAT. THE HEADER AND DATA FIELDS ARE ALL ZEROS, CAUSING THE DEVICE TO USE NORMAL WRITE GATE. THE HEADER AND DATA ARE READ AND COMPARED WITH THE WRITE BUFFER. THE INITIAL SEEK POSITIONS THE HEAD SUCH THAT THERE IS NO IMPLIED SEEK.

FORMAT ZEROS - 16

THIS TEST IS THE SAME AS THE PREVIOUS TEST, EXCEPT THAT DATA IS WRITTEN IN 16 BIT FORMAT.

ZERO FILL TEST

THE TEST EXECUTES A SEEK TO CYLINDER 0 TO INSURE THAT THERE IS NO HEAD MOTION DURING DATA TRANSFER. THIS IS FOLLOWED BY A WRITE HEADER AND DATA COMMAND WITH THE WORD COUNT EQUAL TO THE SIZE OF THE HEADER WHICH CAUSES THE RH70 TO ZERO FILL THE DATA FIELD. THE READ HEADER AND DATA COMMAND THAT FOLLOWS READS A FULL SECTOR AND VERIFIES THAT DATA WAS ZERO FILLED.

FORMAT CHECK ZEROS

THE TEST WRITES HEADER AND AN ALL ZEROS DATA FIELD, THEN PERFORMS A WRITE CHECK HEADER AND DATA COMMAND AND VERIFIES THERE ARE NO ERRORS.

FORMAT CHECK ZEROS W/ WCE ERROR

THE TEST WRITES HEADER AND AN ALL ZEROS DATA FIELD. AFTER COMPLEMENTING THE LAST WORD OF THE WRITE BUFFER, THE TEST PERFORMS A WRITE CHECK HEADER AND DATA COMMAND AND VERIFIES THAT THE CORRECT WRITE CHECK ERROR IS DETECTED.

FORMAT ONES

THE TEST WRITES HEADER AND AN ALL ONES DATA FIELD, THEN READS THE HEADER AND DATA, VERIFYING THE READ BUFFER WITH THE WRITE BUFFER. THE ALL ONES FIELD IS A CONSTANT FREQUENCY, AND THE DRIVE SHOULD USE NORMAL WRITE GATE.

FORMAT CHECK ONES

THE TEST FORMATS AN ALL ONES DATA FIELD, THE PERFORMS A WRITE CHECK HEADER AND DATA COMMAND, VERIFYING THAT THERE ARE NO ERRORS.

FORMAT CHECK ONES W/ WCE ERROR

THE TEST FORMATS AN ALL ONES DATA FIELD, THEN COMPLEMENTS THE LAST WORD OF THE WRITE BUFFER. A WRITE CHECK HEADER AND DATA COMMAND IS EXECUTED, AND THE TEST VERIFIES THAT THE CORRECT WRITE CHECK ERROR IS DETECTED.

FORMAT MULTIPLE SECTORS

THE TEST SEEKS TO CYLINDER 0 TO INSURE THERE IS NO HEAD MOTION DURING DATA TRANSFER. THE WRITE HEADER AND DATA COMMAND FOLLOWS, WITH THE WORD COUNT EQUAL TO MULTIPLE SECTORS. THE WORD COUNT DURING THE READ HEADER AND DATA PORTION OF THE TEST IS SET FOR ALL OF THE FIRST SECTOR AND THE HEADER OF THE SECOND SECTOR.

FORMAT WITH HEAD SWITCHING

THE TEST SEEKS TO CYLINDER 0 TO INSURE THERE IS NO HEAD MOTION DURING DATA TRANSFER. THE WRITE HEADER AND DATA COMMAND STARTS WITH CYLINDER 0, TRACK 0, SECTOR 31. THE WORD COUNT IS EQUAL TO MULTIPLE SECTORS WHICH CAUSES THE SUBSYSTEM TO SWITCH

```
                                    FROM TRACK 0 TO TRACK 1 AFTER THE FIRST SECTOR IS  WRITTEN.   THE
                                    READ  HEADER  AND  DATA  COMMAND  USES  THE  SAME  WORD COUNT AND
```

STARTING SECTOR.

FORMAT WITH IMPLIED SEEK

THIS TEST SEEKS TO THE LAST CYLINDER PRIOR TO WRITING HEADER AND DATA ON CYLINDER 0. THE EXPLICIT SEEK INSURES THAT THERE WILL BE MAXIMUM HEAD MOTION DURING THE IMPLIED SEEK OF THE WRITE COMMAND. THE SAME OPERATION, INCLUDING THE EXPLICIT SEEK IS REPEATED FOR READ HEADER AND DATA.

FORMAT WITH MIDTRANSFER SEEK

THIS TEST WRITES MULTIPLE SECTORS STARTING WITH CYLINDER 0, TRACK 4, SECTOR 31, CAUSING A MIDTRANSFER SEEK AFTER THE FIRST SECTOR IS WRITTEN. THE SAME SECTORS ARE READ WITH READ HEADER AND DATA COMMAND.

FORMAT EACH SECTOR ADDRESS

HEADERS AND DATA OF EACH SECTOR ON CYLINDER 0, TRACK 0 ARE FORMATTED AND READ WITH THE PROGRAM VERIFYING HEADERS AND DATA.

FORMAT EACH TRACK ADDRESS

THIS TEST FORMATS SECTOR 0 OF EACH TRACK ON CYLINDER 0 AND READS EACH SECTOR WITH THE PROGRAM VERIFYING HEADERS AND DATA.

FORMAT PRIME CYLINDERS

THIS TEST FORMATS AND READS SECTOR 0, TRACK 0 ON EACH PRIME CYLINDER, I.E., CYLINDERS 1,2,4,8,...,512.

PAGE 32

FORMAT LAST SECTOR

THIS TEST READS HEADER AND DATA ON THE LAST SECTOR, I.E., CYLINDER 822, TRACK 4, SECTOR 31, AND VERIFIES THAT LBT STATUS SETS.

FOR W/ AOE ERROR

THIS TEST READS MULTIPLE SECTORS STARTING WITH THE LAST SECTOR AND VERIFIES THAT AOE STATUS SETS.

FORMAT INVALID SECTOR ADDRESS

THIS TEST USES AN ILLEGAL SECTOR ADDRESS AND VERIFIES THAT IAE STATUS SETS.

FORMAT INVALID TRACK ADDRESS

THIS TEST USES AN ILLEGAL TRACK ADDRESS AND VERIFIES THAT IAE STATUS SETS.

FORMAT INVALID CYLINDER ADDRESS

THIS TEST USES AN ILLEGAL CYLINDER ADDRESS AND VERIFIES THAT IAE STATUS SETS.

FORMAT AT OFFSET

THE PROGRAM SETS OFFSET MODE AND EXECUTES A WRITE HEADER AND DATA COMMAND, VERIFYING THAT OFFSET MODE IS RESET BY THE WRITE COMMAND.

IVC FORMAT TEST

VOLUME VALID IS RESET BY SETTING AND RESETTING DIAGNOSTIC MODE. THE TEST THEN EXECUTES A WRITE HEADER AND DATA COMMAND AND VERIFIES THAT INVALID COMMAND STATUS SETS. THE TEST IS REPEATED FOR READ HEADER AND DATA COMMAND.

FORMAT ERROR TEST(18 AND 16)

A SINGLE SECTOR IS FORMATTED WITH THE OFFSET REGISTER SET FOR 16 BIT FORMAT AFTER WHICH THE SAME SECTOR IS READ IN 18 BIT FORMAT WITH THE PROGRAM VERIFYING THAT FORMAT ERROR STATUS IS SET. THE SAME PROCEDURE IS REPEATED WITH THE SECTOR WRITTEN IN 18 BIT FORMAT AND READ IN 16 BIT FORMAT.

FORMAT HCE (FIRST AND SECOND HEADER WORDS)

THESE TWO TESTS WRITE AN INCORRECT HEADER THEN READ THE HEADER AND VERIFY THAT THE CORRECT HEADER ERROR IS DETECTED. THE TESTS SETUP THE CORRECT HEADER, THEN COMPLEMENT BIT 0 AND USE THE MODIFIED BUFFER TO WRITE THE HEADER. THE PROCESS IS REPEATED UNTIL EACH BIT POSITION HAS BEEN SEPARATELY TESTED.

```
   806                                  \
```

```
                                ;PROGRAM REVISION #001
                                
                                .TITLE  DZRMDA - RM03 FUNCTIONAL TEST, PART 2
                                ;*COPYRIGHT (C) 1977
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY DOUG RIIKONEN
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*
000001                          $TN=1
                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        14            LOOP ON TEST
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        12            ENABLE EXTENDED STATUS
                                ;*        11            INHIBIT ITERATIONS
                                ;*        10            BELL ON ERROR
                                ;*         9            LOOP ON ERROR
                                ;*         8            LOOP ON TEST IN SWR<7:0>
                                ;*         7            TN128
                                ;*         6            TN64
                                ;*         5            TN32
                                ;*         4            TN16
                                ;*         3            TN8
                                ;*         2            TN4
                                ;*         1            TN2
                                ;*         0            TN1
                                .SBTTL  BASIC DEFINITIONS
                                
                                ;*INITIAL ADDRESS OF T   STACK POINTER *** 1100 ***
001100                          STACK=  1100
                                .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                                .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL
                                
                                ;*MISCELLANEOUS DEFINITIONS
000011                          HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                          LF=     12              ;;CODE FOR LINE FEED
000015                          CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                          PS=     177776          ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
                                
                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                          R0=     %0              ;;GENERAL REGISTER
000001                          R1=     %1              ;;GENERAL REGISTER
000002                          R2=     %2              ;;GENERAL REGISTER
```

```
                                R3=     %3              ;;GENERAL REGISTER
                                R4=     %4              ;;GENERAL REGISTER
                                R5=     %5              ;;GENERAL REGISTER
                                R6=     %6              ;;GENERAL REGISTER
                                R7=     %7              ;;GENERAL REGISTER
                                SP=     %6              ;;STACK POINTER
                                PC=     %7              ;;PROGRAM COUNTER

                                ;*PRIORITY LEVEL D  INITIONS
                                PR0=    0               ;;PRIORITY LEVEL 0
                                PR1=    40              ;;PRIORITY LEVEL 1
                                PR2=    100             ;;PRIORITY LEVEL 2
                                PR3=    140             ;;PRIORITY LEVEL 3
                                PR4=    200             ;;PRIORITY LEVEL 4
                                PR5=    240             ;;PRIORITY LEVEL 5
                                PR6=    300             ;;PRIORITY LEVEL 6
                                PR7=    340             ;;PRIORITY LEVEL 7

                                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
                                SW15=   100000
                                SW14=   40000
                                SW13=   20000
                                SW12=   10000
                                SW11=   4000
                                SW10=   2000
                                SW09=   1000
                                SW08=   400
                                SW07=   200
                                SW06=   100
                                SW05=   40
                                SW04=   20
                                SW03=   10
                                SW02=   4
                                SW01=   2
                                SW00=   1
                                .EQUIV  SW09,SW9
                                .EQUIV  SW08,SW8
                                .EQUIV  SW07,SW7
                                .EQUIV  SW06,SW6
                                .EQUIV  SW05,SW5
                                .EQUIV  SW04,SW4
                                .EQUIV  SW03,SW3
                                .EQUIV  SW02,SW2
                                .EQUIV  SW01,SW1
                                .EQUIV  SW00,SW0

                                ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                                BIT15=  100000
                                BIT14=  40000
                                BIT13=  20000
                                BIT12=  10000
                                BIT11=  4000
                                BIT10=  2000
                                BIT09=  1000
                                BIT08=  400
                                BIT07=  200
```

```
                        BIT06=  100
                        BIT05=  40
                        BIT04=  20
                        BIT03=  10
                        BIT02=  4
                        BIT01=  2
                        BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;"T" BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;"TRAP" TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR

                        .SBTTL  RM03 REGISTER BIT DEFINITIONS

                        ;RMCS1  CONTROL STATUS REGISTER

004000                  DVA     =       BIT11           ;DEVICE AVAILABLE-READ ONLY
000040                  F4      =       BIT05           ;FUNCTION CODE
000020                  F3      =       BIT04           ;FUNCTION CODE
000010                  F2      =       BIT03           ;FUNCTION CODE
000004                  F1      =       BIT02           ;FUNCTION CODE
000002                  F0      =       BIT01           ;FUNCTION CODE
000001                  GO      =       BIT00           ;GO BIT
000077                  FNCMSK  =       000077          ;FUNCTION CODE MASK

                        ;FUNCTION CODES (BITS 01-05 OF RMCS1)

000000                  NOP     =       000000          ;NOP COMMAND
000002                  ILF02   =       000002          ;ILLEGAL COMMAND
000004                  SEEK    =       000004          ;SEEK COMMAND
000006                  RECAL   =       000006          ;RECALIBRATE COMMAND
000010                  DRVCLR  =       000010          ;DRIVE CLEAR COMMAND
000012                  RLEASE  =       000012          ;RELEASE COMMAND
000014                  OFFSET  =       000014          ;OFFSET COMMAND
000016                  RTC     =       000016          ;RETURN TO CENTERLINE COMMAND
000020                  RIP     =       000020          ;READ IN PRESET COMMAND
```

```
000022          PAKACK  =       000022          ;PACK ACKNOWLEDGE COMMAND
000022          PACACK  =       PAKACK
000024          ILF24   =       000024          ;ILLEGAL COMMAND
000026          ILF26   =       000026          ;ILLEGAL COMMAND
000030          SEARCH  =       000030          ;SEARCH COMMAND
000030          ILF30   =       000030          ;ILLEGAL COMMAND
000032          ILF32   =       000032          ;ILLEGAL COMMAND
000034          ILF34   =       000034          ;ILLEGAL COMMAND
000036          ILF36   =       000036          ;ILLEGAL COMMAND
000040          ILF40   =       000040          ;ILLEGAL COMMAND
000042          ILF42   =       000042          ;ILLEGAL COMMAND
000044          ILF44   =       000044          ;ILLEGAL COMMAND
000046          ILF46   =       000046          ;ILLEGAL COMMAND
000050          WCD     =       000050          ;WRITE CHECK DATA COMMAND
000052          WCH     =       000052          ;WRITE CHECK HEADER AND DATA
000054          ILF54   =       000054          ;ILLEGAL COMMAND
000056          ILF56   =       000056          ;ILLEGAL COMMAND
000060          WD      =       000060          ;WRITE DATA COMMAND
000062          WH      =       000062          ;WRITE HEADER AND DATA COMMAND
000064          ILF64   =       000064          ;ILLEGAL COMMAND
000066          ILF66   =       000066          ;ILLEGAL COMMAND
000070          RD      =       000070          ;READ DATA COMMAND
000072          RH      =       000072          ;READ HEADER AND DATA COMMAND
000074          ILF74   =       000074          ;ILLEGAL COMMAND
000076          ILF76   =       000076          ;ILLEGAL COMMAND

                ;RMDA   DISK ADDRESS REGISTER

002000          TA4     =       BIT10           ;TRACK ADDRESS 4
001000          TA2     =       BIT09           ;TRACK ADDRESS 2
000400          TA1     =       BIT08           ;TRACK ADDRESS 1
000020          SA16    =       BIT04           ;SECTOR ADDRESS 16
000010          SA8     =       BIT03           ;SECTOR ADDRESS 8
000004          SA4     =       BIT02           ;SECTOR ADDRESS 4
000002          SA2     =       BIT01           ;SECTOR ADDRESS 2
000001          SA1     =       BIT00           ;SECTOR ADDRESS 1

                ;TRACK,SECTOR MASKS

003400          TAMSK   =       003400          ;TRACK ADDRESS MASK
000037          SAMSK   =       000037          ;SECTOR ADDRESS MASK

                ;RMDS   DRIVE STATUS REGISTER

100000          ATA     =       BIT15           ;ATTENTION ACTIVE
040000          ERR     =       BIT14           ;COMPOSITE ERROR
020000          PIP     =       BIT13           ;POSITIONING IN PROGRESS
010000          MOL     =       BIT12           ;MEDIUM ON LINE
004000          WRL     =       BIT11           ;WRITE LOCK
002000          LBT     =       BIT10           ;LAST BLOCK TRANSFERRED
001000          PGM     =       BIT09           ;PROGRAMMABLE
000400          DPR     =       BIT08           ;DRIVE PRESENT
000200          DRY     =       BIT07           ;DRIVE READY
000100          VV      =       BIT06           ;VOLUME VALID
000001          OM      =       BIT00           ;OFFSET MODE ACTIVE
```

```
                                ;RMER1  ERROR REGISTER #1

100000          DCK     =       BIT15           ;DATA CHECK ERROR
040000          UNS     =       BIT14           ;DRIVE UNSAFE
020000          OPI     =       BIT13           ;OPERATION INCOMPLETE
010000          DTE     =       BIT12           ;DRIVE TIMING ERROR
004000          WLE     =       BIT11           ;WRITE LOCK ERROR
002000          IAE     =       BIT10           ;INVALID ADDRESS ERROR
001000          AOE     =       BIT09           ;ADDRESS OVERFLOW ERROR
000400          HCRC    =       BIT08           ;HEADER CRC ERROR
000200          HCE     =       BIT07           ;HEADER COMPARE ERROR
000100          ECH     =       BIT06           ;ECC "HARD" ERROR
000040          WCF     =       BIT05           ;WRITE CLOCK FAILURE
000020          FER     =       BIT04           ;FORMAT ERROR
000010          PAR     =       BIT03           ;PARITY ERROR
000004          RMR     =       BIT02           ;REGISTER MODIFICATION REFUSED
000002          ILR     =       BIT01           ;ILLEGAL REGISTER
000001          ILF     =       BIT00           ;ILLEGAL FUNCTION

115760          NDTMSK  =       DCK!DTE!WLE!AOE!HCRC!HCE!ECH!WCF!FER
                ;"NDTMSK" IS USED TO MASK ERROR REGISTER 1 DURING NON - DATA
                ;COMMANDS, I.E., HOUSEKEEPING AND POSITIONING COMMANDS

                ;RMAS   ATTENTION SUMMARY REGISTER

000377          ATNMSK  =       377             ;MASK FOR ATTENTION BITS

                ;RMLA   LOOK AHEAD REGISTER

002000          SC4     =       BIT10           ;SECTOR COUNT = 16
001000          SC3     =       BIT09           ;SECTOR COUNT = 8
000400          SC2     =       BIT08           ;SECTOR COUNT = 4
000200          SC1     =       BIT07           ;SECTOR COUNT = 2
000100          SC0     =       BIT06           ;SECTOR COUNT = 1

003700          SCTMSK  =       003700          ;SECTOR COUNT MASK

                ;RMMR   MAINTENANCE REGISTER

                ;       WRITE ONLY BITS

100000          DBCK    =       BIT15           ;DEBUG CLOCK
040000          DBEN    =       BIT14           ;DEBUG CLOCK ENABLE
020000          DEBL    =       BIT13           ;DIAGNOSTIC END OF BLOCK
010000          DTO     =       BIT12           ;DIAGNOSTIC TIMEOUT
004000          MCLK    =       BIT11           ;MAINTENANCE CLOCK
002000          MRD     =       BIT10           ;READ DATA
001000          MUR     =       BIT09           ;UNIT READY
000400          MOC     =       BIT08           ;ON CYLINDER
000200          MSER    =       BIT07           ;SEEK ERROR
000100          MDF     =       BIT06           ;DRIVE FAULT
000040          MS      =       BIT05           ;SECTOR PULSE
000010          MWP     =       BIT03           ;WRITE PROTECT
000004          MI      =       BIT02           ;INDEX PULSE
000002          MSC     =       BIT01           ;SECTOR COMPARE
000001          DMD     =       BIT00           ;DIAGNOSTIC MODE
```

```
                                ;       READ ONLY BITS
                                ;
100000          OCC     =       BIT15           ;OCCUPIED
040000          RG      =       BIT14           ;RUN AND GO
020000          EBL     =       BIT13           ;END OF BLOCK
010000          REX     =       BIT12           ;EXCEPTION
004000          ESRC    =       BIT11           ;ENABLE SEARCH
002000          PLFS    =       BIT10           ;LOOKING FOR SYNC
001000          ECRC    =       BIT09           ;ENABLE CRC OUT
000400          PDA     =       BIT08           ;DATA AREA
000200          PHA     =       BIT07           ;HEADER AREA
000100          CONT    =       BIT06           ;CONTINUE
000040          WC      =       BIT05           ;WORD CLOCK
000020          EECC    =       BIT04           ;ENABLE ECC OUT
000010          MWD     =       BIT03           ;WRITE DATA BIT
000004          LS      =       BIT02           ;LAST SECTOR
000002          LST     =       BIT01           ;LAST SECTOR AND TRACK
000001          DMD     =       BIT00           ;DIAGNOSTIC MODE

                                ;RMDT   DRIVE TYPE REGISTER

100000          NSA     =       BIT15           ;NOT SECTOR ADDRESSED=0
040000          TAP     =       BIT14           ;TAPE DRIVE = 0
020000          MOH     =       BIT13           ;MOVING HEAD = 1
004000          DRQ     =       BIT11           ;DRIVE REQUEST REQUIRED

020024          SNGPRT  =       020024          ;SINGLE PORT DRIVE TYPE
024024          DULPRT  =       024024          ;DUAL PORT DRIVE TYPE

                                ;RMOF   OFFSET REGISTER

010000          FMT16   =       BIT12           ;16 BIT WORD FORMAT
004000          ECI     =       BIT11           ;ECC INHIBIT
002000          HCI     =       BIT10           ;HEADER COMPARE INHIBIT
000200          OFD     =       BIT07           ;OFFSET FORWARD


                                ;RMDC   DESIRED CYLINDER ADDRESS REGISTER
001777          CYLMSK  =       1777            ;MASK FOR CYLINDER ADDRESS

                                ;RMMR2  MAINTENANCE REGISTER #2

                                ;       READ ONLY BITS
100000          RQA     =       BIT15           ;PORT A REQUEST
040000          RQB     =       BIT14           ;PORT B REQUEST
020000          TAG     =       BIT13           ;TAG CONTROL
010000          TST     =       BIT12           ;COMMAND SEQUENCE TEST BIT
004000          CC      =       BIT11           ;CONTROL OR CYLINDER TAG
002000          CH      =       BIT10           ;CONTROL OR HEAD TAG
001000          BB09    =       BIT09           ;TAG BUS
000400          BB08    =       BIT08           ;TAG BUS
000200          BB07    =       BIT07           ;TAG BUS
000100          BB06    =       BIT06           ;TAG BUS
000040          BB05    =       BIT05           ;TAG BUS
000020          BB04    =       BIT04           ;TAG BUS
```

```
                    000010          BB03    =       BIT03           ;TAG BUS
                    000004          BB02    =       BIT02           ;TAG BUS
                    000002          BB01    =       BIT01           ;TAG BUS
                    000001          BB00    =       BIT00           ;TAG BUS


                                    ;RMER2  ERROR REGISTER 2

                    100000          BSE     =       BIT15           ;BAD SECTOR ERROR
                    040000          SKI     =       BIT14           ;SEEK INCOMPLETE
                    020000          OPE     =       BIT13           ;OPERATOR PLUG ERROR
                    010000          IVC     =       BIT12           ;INVALID COMMAND ERROR
                    004000          LSC     =       BIT11           ;LOSS OF SYSTEM CLOCK
                    002000          LBC     =       BIT10           ;LOSS OF BIT CLOCK
                    000200          DVC     =       BIT07           ;DEVICE CHECK
                    000010          DPE     =       BIT03           ;DATA PARITY ERROR

                                    .SBTTL  PROGRAM MNEMONICS

                    100000          MSE     =       BIT15           ;MANUFACTURING DETECTED SECTOR ERROR
                    040000          USE     =       BIT14           ;USER DETECTED SECTOR ERROR

                                    .SBTTL  RM03 REGISTER INDEX VALUES

                    000000          RMCS1   =       00              ;CONTROL STATUS REGISTER
                    000006          RMDA    =       06              ;DISK ADDRESS REGISTER
                    000012          RMDS    =       12              ;DRIVE STATUS REGISTER
                    000014          RMER1   =       14              ;ERROR REGISTER 1
                    000016          RMAS    =       16              ;ATTENTION SUMMARY REGISTER
                    000020          RMLA    =       20              ;LOOK AHEAD REGISTER
                    000024          RMMR1   =       24              ;MAINTENANCE REGISTER
                    000026          RMDT    =       26              ;DRIVE TYPE REGISTER
                    000030          RMSN    =       30              ;SERIAL NUMBER REGISTER
                    000032          RMOF    =       32              ;OFFSET REGISTER
                    000034          RMDC    =       34              ;DESIRED CYLINDER REGISTER
                    000036          RMCC    =       36              ;CURRENT CYLINDER REGISTER
                    000040          RMMR2   =       40              ;MAINTENANCE REGISTER 2
                    000042          RMER2   =       42              ;ERROR REGISTER 2
                    000044          RMEC1   =       44              ;ECC POSITION REGISTER
                    000046          RMEC2   =       46              ;ECC PATTERN REGISTER
                    000050          ILRG50  =       50              ;ILLEGAL REGISTER 50
                    000052          ILRG52  =       52              ;ILLEGAL REGISTER 52
                    000054          ILRG54  =       54              ;ILLEGAL REGISTER 54
                    000056          ILRG56  =       56              ;ILLEGAL REGISTER 56
                    000060          ILRG60  =       60              ;ILLEGAL REGISTER 60
                    000062          ILRG62  =       62              ;ILLEGAL REGISTER 62
                    000064          ILRG64  =       64              ;ILLEGAL REGISTER 64
                    000066          ILRG66  =       66              ;ILLEGAL REGISTER 66
                    000070          ILRG70  =       70              ;ILLEGAL REGISTER 70
                    000072          ILRG72  =       72              ;ILLEGAL REGISTER 72
                    000074          ILRG74  =       74              ;ILLEGAL REGISTER 74
                    000076          ILRG76  =       76              ;ILLEGAL REGISTER 76


                    000077          IDXMSK  =       77              ;MASK FOR REGISTER INDEX NUMBER

```

```
1199                                    .SBTTL  RH CONTROLLER REGISTER BIT DEFINITIONS
1200
1201                                    ;RMCS1  CONTROL STATUS REGISTER #1
1202
1203            100000                  SC      =       BIT15           ;SPECIAL CONDITION-READ ONLY
1204            040000                  TRE     =       BIT14           ;TRANSFER ERROR
1205            020000                  MCPE    =       BIT13           ;MASSBUS CONTROL BUS PARITY
1206                                                                    ;ERROR-READ ONLY
1207            002000                  PSEL    =       BIT10           ;PORT B SELECT
1208            001000                  A17     =       BIT09           ;ADDRESS EXTENSION
1209            000400                  A16     =       BIT08           ;ADDRESS EXTENSION
1210            000200                  RDY     =       BIT07           ;READY-READ ONLY
1211            000100                  IE      =       BIT06           ;INTERRUPT ENABLE
1212
1213                                    ;RMCS2  RH CONTROL STATUS REGISTER #2
1214
1215            100000                  DLT     =       BIT15           ;DATA LATE-READ ONLY
1216            040000                  WCE     =       BIT14           ;WRITE CHECK ERROR-READ ONLY
1217            020000                  UPE     =       BIT13           ;UNIBUS PARITY ERROR
1218            010000                  NED     =       BIT12           ;NONEXISTANT DRIVE-READ ONLY
1219            004000                  NEM     =       BIT11           ;NONEXISTANT MEMORY-READ ONLY
1220            002000                  PGE     =       BIT10           ;PROGRAM ERROR-READ ONLY
1221            001000                  MXF     =       BIT09           ;MISSED TRANSFER
1222            000400                  MDPE    =       BIT08           ;MASSBUS DATA BUS PARITY
1223                                                                    ;ERROR-READ ONLY
1224            000200                  OR      =       BIT07           ;OUTPUT READY-READ ONLY
1225            000100                  IR      =       BIT06           ;INPUT READY-READ ONLY
1226            000040                  CLR     =       BIT05           ;CONTROLLER CLEAR
1227            000020                  PAT     =       BIT04           ;PARITY TEST
1228            000010                  BAI     =       BIT03           ;UNIBUS ADDRESS INCREMENT
1229                                                                    ;INHIBIT
1230            000004                  U2      =       BIT02           ;UNIT SELECT
1231            000002                  U1      =       BIT01           ;UNIT SELECT
1232            000001                  U0      =       BIT00           ;UNIT SELECT
1233
1234                                    ;UNIT SELECT MASK
1235            000007                  UNTMSK  =       7               ;UNIT SELECT MASK
1236
1237                                    ;RMCS3  RH70 CONTROL STATUS REGISTER #3
1238            100000                  APE     =       BIT15           ;ADDRESS PARITY ERROR
1239            040000                  DPEHI   =       BIT14           ;DATA PARITY ERROR HIGH WORD
1240            020000                  DPELO   =       BIT13           ;DATA PARITY ERROR LOW WORD
1241            010000                  WCEHI   =       BIT12           ;WRITE CHECK ERROR HIGH WORD
1242            004000                  WCELO   =       BIT11           ;WRITE CHECK ERROR LOW WORD
1243            002000                  DBL     =       BIT10           ;DOUBLE WORD TRANSFER
1244            000100                  IE      =       BIT06           ;INTERRUPT ENABLE
1245            000010                  IPCK3   =       BIT03           ;INVERT PARITY CHECK
1246            000004                  IPCK2   =       BIT02           ;INVERT PARITY CHECK
1247            000002                  IPCK1   =       BIT01           ;INVERT PARITY CHECK
1248            000001                  IPCK0   =       BIT00           ;INVERT PARITY CHECK
1249                                    .SBTTL  RH CONTROLLER REGISTER INDEX VALUES
1250
1251            000000                  RMCS1   =       00              ;CONTROL, STATUS REGISTER
1252            000002                  RMWC    =       02              ;WORD COUNT REGISTER
1253            000004                  RMBA    =       04              ;BUS ADDRESS REGISTER
1254            000010                  RMCS2   =       10              ;CONTROLLER STATUS REGISTER
```

```
                000022          RMDB    =       22              ;DATA BUFFER
                000050          RMBAE   =       50              ;BUS ADDRESS EXTENSION
                000052          RMCS3   =       52              ;CONTROL STATUS REGISTER #3

                176700          ABASE   =       176700          ;UNIBUS ADDRESS
                120254          AVECT1  =       120254          ;UNIBUS VECTOR ADDRESS AND PRIORITY


                                .SBTTL  TRAP CATCHER

                000000                  .=0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                000174                  .=174
000174  000000  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER


                                .SBTTL  ACT11 HOOKS

                                ;;******************************************************************
                                ;HOOKS REQUIRED BY ACT11
                000200                  $SVPC=.                 ;SAVE PC
                000046                  .=46
000046  032452                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                000052                  .=52
000052  000000                  .WORD   0               ;;2)SET LOC.52 TO ZERO
                000200                  .=$SVPC                 ;; RESTORE PC


                                .SBTTL  STARTING ADDRESS

                                ;THE PROGRAM STARTS AT LOCATION 200
                000200          .       =       200
000200  000137  005322                  JMP     START           ;JUMP TO START OF PROGRAM

                001100          .=1100
                                .SBTTL  APT PARAMETER BLOCK

                                ;;******************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;******************************************************************
                001100                  .$X=.   ;;SAVE CURRENT LOCATION
                000024                  .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                  200     ;;FOR APT START UP
                000044                  .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  001100                  $APTHDR ;;POINT TO APT HEADER BLOCK
                001100                  .=.$X   ;;RESET LOCATION COUNTER
                                ;;******************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

001100                          $APTHD:
001100  000000  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001102  001222  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001104  000001  $TSTM:  .WORD   1       ;;RUN TIM OF LONGEST TEST
```

APT PARAMETER BLOCK

```
  1311  001106  000002          $PASTM  .WORD   2       ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
  1312  001110  000002          $UNITM: .WORD   2       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
  1313  001112  000042                  .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
  1314          001114          TAGADR  =       .
```

```
                                .SBTTL  COMMON TAGS

                                ;;**********************************************************************
                                ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                                ;*USED IN THE PROGRAM.

                                        .=TAGADR
001114                  $CMTAG:                         ;;START OF COMMON TAGS
001114  000000                  .WORD   0
001116  000                     $TSTNM: .BYTE   0       ;;CONTAINS THE TEST NUMBER
001117  000                     $ERFLG: .BYTE   0       ;;CONTAINS ERROR FLAG
001120  000000                  $ICNT:  .WORD   0       ;;CONTAINS SUBTEST ITERATION COUNT
001122  000000                  $LPADR: .WORD   0       ;;CONTAINS SCOPE LOOP ADDRESS
001124  000000                  $LPERR: .WORD   0       ;;CONTAINS SCOPE RETURN FOR ERRORS
001126  000000                  $ERTTL: .WORD   0       ;;CONTAINS TOTAL ERRORS DETECTED
001130  000                     $ITEMB: .BYTE   0       ;;CONTAINS ITEM CONTROL BYTE
001131  001                     $ERMAX: .BYTE   1       ;;CONTAINS MAX. ERRORS PER TEST
001132  000000                  $ERRPC: .WORD   0       ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001134  000000                  $GDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'GOOD' DATA
001136  000000                  $BDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'BAD' DATA
001140  000000                  $GDDAT: .WORD   0       ;;CONTAINS 'GOOD' DATA
001142  000000                  $BDDAT: .WORD   0       ;;CONTAINS 'BAD' DATA
001144  000000                          .WORD   0       ;;RESERVED--NOT TO BE USED
001146  000000                          .WORD   0
001150  000                     $AUTOB: .BYTE   0       ;;AUTOMATIC MODE INDICATOR
001151  000                     $INTAG: .BYTE   0       ;;INTERRUPT MODE INDICATOR
001152  000000                          .WORD   0
001154  177570                  SWR:    .WORD   DSWR    ;;ADDRESS OF SWITCH REGISTER
001156  177570                  DISPLAY: .WORD  DDISP   ;;ADDRESS OF DISPLAY REGISTER
001160  177560                  $TKS:   177560          ;;TTY KBD STATUS
001162  177562                  $TKB:   177562          ;;TTY KBD BUFFER
001164  177564                  $TPS:   177564          ;;TTY PRINTER STATUS REG. ADDRESS
001166  177566                  $TPB:   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
001170  000                     $NULL:  .BYTE   0       ;;CONTAINS NULL CHARACTER FOR FILLS
001171  002                     $FILLS: .BYTE   2       ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001172  012                     $FILLC: .BYTE   12      ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001173  000                     $TPFLG: .BYTE   0       ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001174  000000                  $TMP0:  .WORD   0       ;;USER DEFINED
001176  000000                  $TMP1:  .WORD   0       ;;USER DEFINED
001200  000000                  $TMP2:  .WORD   0       ;;USER DEFINED
001202  000000                  $TMP3:  .WORD   0       ;;USER DEFINED
001204  000000                  $TMP4:  .WORD   0       ;;USER DEFINED
001206  000000                  $TIMES: 0               ;;MAX. NUMBER OF ITERATIONS
001210  000000                  $ESCAPE:0               ;;ESCAPE ON ERROR ADDRESS
001212  177607  000377           $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001216  077                     $QUES:  .ASCII  /?/     ;;QUESTION MARK
001217  015                     $CRLF:  .ASCII  <15>    ;;CARRIAGE RETURN
001220  000012                  $LF:    .ASCIZ  <12>    ;;LINE FEED
                                ;;**********************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;**********************************************************************
                                .EVEN
001222                  $MAIL:                          ;;APT MAILBOX
001222  000000                  $MSGTY: .WORD   AMSGTY   ;;MESSAGE TYPE CODE
001224  000000                  $FATAL: .WORD   AFATAL   ;;FATAL ERROR NUMBER
```

```
1371  001226  000000          $TESTN: .WORD   ATESTN  ;;TEST NUMBER
1372  001230  000000          $PASS:  .WORD   APASS   ;;PASS COUNT
1373  001232  000000          $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
1374  001234  000000          $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
1375  001236  000000          $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
1376  001240  000000          $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
1377  001242                  $ETABLE:                ;;APT ENVIRONMENT TABLE
1378  001242     000          $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
1379  001243     000          $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
1380  001244  000000          $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
1381  001246  000000          $USWR:  .WORD   AUSWR   ;;USER SWITCHES
1382  001250  000000          $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
1383                          ;*                      BITS 15-11=CPU TYPE
1384                          ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1385                          ;*                              11/70=06,PDQ=07,Q=10
1386                          ;*                      BIT 10=REAL TIME CLOCK
1387                          ;*                      BIT  9=FLOATING POINT PROCESSOR
1388                          ;*                      BIT  8=MEMORY MANAGEMENT
1389  001252     000          $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
1390  001253     000          $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
1391                          ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
1392                          ;*                              900 NSEC CORE=001
1393                          ;*                              300 NSEC BIPOLAR=002
1394                          ;*                              500 NSEC MOS=003
1395  001254  000000          $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
1396                          ;*                       MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
1397  001256     000          $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
1398  001257     000          $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
1399  001260  000000          $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
1400  001262     000          $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
1401  001263     000          $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
1402  001264  000000          $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
1403  001266     000          $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
1404  001267     000          $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
1405  001270  000000          $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
1406  001272  120254          $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
1407  001274  000000          $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
1408  001276  176700          $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
1409  001300  000000          $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
1410  001302  000000          $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
1411  001304  000000          $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
1412  001306  000000          $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
1413  001310  000000          $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
1414  001312  000000          $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
1415  001314  000000          $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
1416  001316  000000          $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
1417  001320  000000          $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
1418  001322  000000          $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
1419  001324  000000          $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
1420  001326                  $ETEND:
1421                          .MEXIT
1422
1423                          ;THE REGISTER INPUT BUFFER IS USED FOR
1424                          ;STORING DRIVE STATUS
1425
1426  001326                  GETBUF:
```

```
                                ;REGISTER INPUT BUFFER
001326  000000                  RMCS1I: .WORD                   ;CONTROL,STATUS REGISTER
001330  000000                  RMWCI: .WORD                    ;WORD COUNT REGISTER
001332  000000                  RMBAI: .WORD                    ;BUS ADDRESS REGISTER
001334  000000                  RMDAI: .WORD                    ;DISK ADDRESS REGISTER
001336  000000                  RMCS2I: .WORD                   ;CONTROLLER STATUS REGISTER
001340  000000                  RMDSI: .WORD                    ;DRIVE STATUS REGISTER
001342  000000                  RMER1I: .WORD                   ;ERROR REGISTER 1
001344  000000                  RMASI: .WORD                    ;ATTENTION SUMMARY REGISTER
001346  000000                  RMLAI: .WORD                    ;LOOK AHEAD REGISTER
001350  000000                  RMDBI: .WORD                    ;DATA BUFFER
001352  000000                  RMMR1I: .WORD                   ;MAINTENANCE REGISTER #1
001354  000000                  RMDTI: .WORD                    ;DRIVE TYPE REGISTER
001356  000000                  RMSNI: .WORD                    ;SERIAL NUMBER REGISTER
001360  000000                  RMOFI: .WORD                    ;OFFSET REGISTER
001362  000000                  RMDCI: .WORD                    ;DESIRED CYLINDER REGISTER
001364  000000                  RMCCI: .WORD                    ;CURRENT CYLINDER REGISTER
001366  000000                  RMMR2I: .WORD                   ;MAINTENANCE REGISTER #2
001370  000000                  RMER2I: .WORD                   ;ERROR REGISTER 2
001372  000000                  RMEC1I: .WORD                   ;ECC POSITION REGISTER
001374  000000                  RMEC2I: .WORD                   ;ECC PATTERN REGISTER

                                ;THE REGISTER OUTPUT BUFFER IS USED FOR
                                ;ASSEMBLING DATA GOING TO REGISTER

001376                          PUTBUF:

                                ;REGISTER OUTPUT BUFFER
001376  000000                  RMCS1O: .WORD                   ;CONTROL,STATUS REGISTER
001400  000000                  RMWCO: .WORD                    ;WORD COUNT REGISTER
001402  000000                  RMBAO: .WORD                    ;BUS ADDRESS REGISTER
001404  000000                  RMDAO: .WORD                    ;DISK ADDRESS REGISTER
001406  000000                  RMCS2O: .WORD                   ;CONTROLLER STATUS REGISTER
001410  000000                  RMDSO: .WORD                    ;DRIVE STATUS REGISTER
001412  000000                  RMER1O: .WORD                   ;ERROR REGISTER 1
001414  000000                  RMASO: .WORD                    ;ATTENTION SUMMARY REGISTER
001416  000000                  RMLAO: .WORD                    ;LOOK AHEAD REGISTER
001420  000000                  RMDBO: .WORD                    ;DATA BUFFER
001422  000000                  RMMR1O: .WORD                   ;MAINTENANCE REGISTER #1
001424  000000                  RMDTO: .WORD                    ;DRIVE TYPE REGISTER
001426  000000                  RMSNO: .WORD                    ;SERIAL NUMBER REGISTER
001430  000000                  RMOFO: .WORD                    ;OFFSET REGISTER
001432  000000                  RMDCO: .WORD                    ;DESIRED CYLINDER REGISTER
001434  000000                  RMCCO: .WORD                    ;CURRENT CYLINDER REGISTER
001436  000000                  RMMR2O: .WORD                   ;MAINTENANCE REGISTER #2
001440  000000                  RMER2O: .WORD                   ;ERROR REGISTER 2
001442  000000                  RMEC1O: .WORD                   ;ECC POSITION REGISTER
001444  000000                  RMEC2O: .WORD                   ;ECC PATTERN REGISTER

                                ;EACH WORD OF THE TEST QUE CONTAINS THE DEVICE NUMBER IN
                                ;THE LOW BYTE AND THE ATTENTION BIT IN THE HIGH BYTE. A ZERO
                                ;WORD IS A BLANK AND REPRESENTS THE END OF THE QUE.

001446  000012                  TSTQUE: .BLKW   10.      ;TEST QUE
```

```
1483                                    ;MEDIA ENABLE IS SET IF THE BAD SECTOR FILES HAVE BEEN RECOVERED
1484                                    ;FOR THE UNIT UNDER TEST, OTHERWISE IT IS ZERO.
1485  001472  000000                    MEDENB: .WORD           ;MEDIA ENABLE
1486
1487                                    ;LOCATIONS "ASNDC" AND "ASNDC" CONTAIN THE CYLINDER, TRACK AND SECTOR
1488                                    ;ADDRESS ASSIGNED BY THE BAD SECTOR MODULE.
1489  001474  000000                    ASNDC:  .WORD           ;ASSIGNED DESIRED CYLINDER
1490  001476  000000                    ASNDA:  .WORD           ;ASSIGNED TRACK, AND SECTOR
1491  001500  000000                    CLKADR: .WORD           ;UNIBUS ADDRESS OF KW11 CLOCK
1492  001502  000000                    CLKVCT: .WORD           ;VECTOR ADDRESS OF KW11 CLOCK
1493
1494                                    ;THE GET INDEX TABLE CONTAINS A BYTE LIST OF REGISTERS WHICH
1495                                    ;ARE READ BY THE GET SUBROUTINE.  THE LIST IS TERMINATED BY
1496                                    ;A NEGATIVE BYTE.
1497  001504  000027                    GETINX: .BLKB   23.             ;GET INDEX TABLE
1498
1499                                    ;THE PUT INDEX TABLE ICONTAINS A BYTE LIST OF REGISTERS WHICH
1500                                    ;ARE WRITTEN BY THE PUT SUBROUTINE.  THE LIST IS TERMINATED BY
1501                                    ;A NEGATIVE BYTE.
1502  001533  000027                    PUTINX: .BLKB   23.             ;PUT INDEX TABLE
1503
1504                                    ;PUT TAGS HERE
1505
```

```
                        .SBTTL  ERROR POINTER TABLE

                        ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                        ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                        ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                        ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                        ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                        ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                        ;*      DH              ;;POINTS TO THE DATA HEADER
                        ;*      DT              ;;POINTS TO THE DATA
                        ;*      DF              ;;POINTS TO THE DATA FORMAT

001562                  $ERRTB:
```

```
1523
1524  001562  064434                ;ERROR  1       WRONG UNIT SELECTED
1525  001564  070526                        EMT1
1526  001566  070652                        EHT1
1527  001570  070742                        EDT1
1528                                  EFT1
1529
1530                                ;ERROR  2       DEVICE WENT UNAVAILABLE
1531  001572  064440                        EMT2
1532  001574  070526                        EHT1
1533  001576  070652                        EDT1
1534  001600  070742                        EFT1
1535
1536
1537                                ;ERROR  3       DEVICE WENT NONEXISTENT
1538  001602  064446                        EMT3
1539  001604  070526                        EHT1
1540  001606  070652                        EDT1
1541  001610  070742                        EFT1
1542
1543
1544                                ;ERROR  4       CONTROLLER NOT READY
1545  001612  064454                        EMT4
1546  001614  070526                        EHT1
1547  001616  070652                        EDT1
1548  001620  070742                        EFT1
1549
1550
1551                                ;ERROR  5       DRIVE NOT READY AND GO NOT RESET
1552  001622  064462                        EMT5
1553  001624  070526                        EHT1
1554  001626  070652                        EDT1
1555  001630  070742                        EFT1
1556
1557
1558                                ;ERROR  6       UNEXPECTED VALUE FOR "ATA" STATUS
1559  001632  064470                        EMT6
1560  001634  070526                        EHT1
1561  001636  070652                        EDT1
1562  001640  070742                        EFT1
1563
1564
1565                                ;ERROR  7       BUS TIMEOUT TRYING TO READ OR WRITE REGISTER
1566  001642  064476                        EMT7
1567  001644  000000                        0
1568  001646  000000                        0
1569  001650  000000                        0
1570
1571
1572                                ;ERROR  10      DRIVE NOT READY BUT GO IS RESET
1573  001652  064504                        EMT10
1574  001654  070526                        EHT1
1575  001656  070652                        EDT1
1576  001660  070742                        EFT1
1577
1578
```

ERROR POINTER TABLE

```
1579
1580  001662  064510                ;ERROR  11      GO NOT RESET BUT DRIVE IS READY
1581  001664  070526                        EMT11
1582  001666  070652                        EHT1
1583  001670  070742                        EDT1
1584                                  EFT1
1585
1586                                ;ERROR  12      INCORRECT FUNCTION CODE
1587  001672  064514                        EMT12
1588  001674  070526                        EHT1
1589  001676  070652                        EDT1
1590  001700  070742                        EFT1
1591
1592
1593                                ;ERROR  13      PARITY ERROR READING REMOTE REGISTERS
1594  001702  064522                        EMT13
1595  001704  070526                        EHT1
1596  001706  070652                        EDT1
1597  001710  070742                        EFT1
1598
1599
1600                                ;ERROR  14      TRANSFER ERROR IS INCORRECT
1601  001712  064534                        EMT14
1602  001714  070526                        EHT1
1603  001716  070652                        EDT1
1604  001720  070742                        EFT1
1605
1606
1607                                ;ERROR  15      INCORRECT WORD COUNT
1608  001722  064542                        EMT15
1609  001724  070526                        EHT1
1610  001726  070652                        EDT1
1611  001730  070742                        EFT1
1612
1613
1614                                ;ERROR  16      INCORRECT BUS ADDRESS
1615  001732  064556                        EMT16
1616  001734  070526                        EHT1
1617  001736  070652                        EDT1
1618  001740  070742                        EFT1
1619
1620
1621                                ;ERROR  17      INCORRECT LBT STATUS
1622  001742  064560                        EMT17
1623  001744  070526                        EHT1
1624  001746  070652                        EDT1
1625  001750  070742                        EFT1
1626
1627
1628                                ;ERROR  20      INCORRECT AOE
1629  001752  064570                        EMT20
1630  001754  070526                        EHT1
1631  001756  070652                        EDT1
1632  001760  070742                        EFT1
1633
1634
```

```
1635                                    ;ERROR  21      INCORRECT DISK ADDRESS
1636  001762  064600                              EMT21
1637  001764  070526                              EHT1
1638  001766  070652                              EDT1
1639  001770  070742                              EFT1
1640
1641
1642                                    ;ERROR  22      INCORRECT CYLINDER ADDRESS
1643  001772  064610                              EMT22
1644  001774  070526                              EHT1
1645  001776  070652                              EDT1
1646  002000  070742                              EFT1
1647
1648
1649                                    ;ERROR  23      INCORRECT WLE STATUS
1650  002002  064620                              EMT23
1651  002004  070526                              EHT1
1652  002006  070652                              EDT1
1653  002010  070742                              EFT1
1654
1655
1656                                    ;ERROR  24      INCORRECT UPE STATUS
1657  002012  064630                              EMT24
1658  002014  070526                              EHT1
1659  002016  070652                              EDT1
1660  002020  070742                              EFT1
1661
1662
1663                                    ;ERROR  25      INCORRECT WCF STATUS
1664  002022  064640                              EMT25
1665  002024  070526                              EHT1
1666  002026  070652                              EDT1
1667  002030  070742                              EFT1
1668
1669
1670                                    ;ERROR  26      INCORRECT WCE STATUS
1671  002032  064650                              EMT26
1672  002034  070526                              EHT1
1673  002036  070652                              EDT1
1674  002040  070742                              EFT1
1675
1676
1677                                    ;ERROR  27      INCORRECT MDPE STATUS
1678  002042  064660                              EMT27
1679  002044  070526                              EHT1
1680  002046  070652                              EDT1
1681  002050  070742                              EFT1
1682
1683
1684                                    ;ERROR  30      INCORRECT DCK STATUS
1685  002052  064670                              EMT30
1686  002054  070526                              EHT1
1687  002056  070652                              EDT1
1688  002060  070742                              EFT1
1689
1690
```

```
1691
1692  002062  064700                ;ERROR  31      INCORRECT ECH STATUS
1693  002064  070526                        EMT31
1694  002066  070652                        EHT1
1695  002070  070742                        EDT1
1696                                  EFT1
1697
1698                                ;ERROR  32      DLT SHOULD NOT BE SET
1699  002072  064710                        EMT32
1700  002074  070526                        EHT1
1701  002076  070652                        EDT1
1702  002100  070742                        EFT1
1703
1704
1705                                ;ERROR  33      MXF SHOULD NOT BE SET
1706  002102  064720                        EMT33
1707  002104  070526                        EHT1
1708  002106  070652                        EDT1
1709  002110  070742                        EFT1
1710
1711
1712                                ;ERROR  34      DTE SHOULD NOT BE SET
1713  002112  064730                        EMT34
1714  002114  070526                        EHT1
1715  002116  070652                        EDT1
1716  002120  070742                        EFT1
1717
1718
1719                                ;ERROR  35      INCORRECT HCRC STATUS
1720  002122  064740                        EMT35
1721  002124  070526                        EHT1
1722  002126  070652                        EDT1
1723  002130  070742                        EFT1
1724
1725
1726                                ;ERROR  36      INCORRECT HCE STATUS
1727  002132  064750                        EMT36
1728  002134  070526                        EHT1
1729  002136  070652                        EDT1
1730  002140  070742                        EFT1
1731
1732
1733                                ;ERROR  37      INCORRECT FER STATUS
1734  002142  064760                        EMT37
1735  002144  070526                        EHT1
1736  002146  070652                        EDT1
1737  002150  070742                        EFT1
1738
1739
1740                                ;ERROR  40      DPE SHOULD NOT BE SET (NOT A DATA COMMAND)
1741  002152  064770                        EMT40
1742  002154  070526                        EHT1
1743  002156  070652                        EDT1
1744  002160  070742                        EFT1
1745
1746
```

```
1747                                    ;ERROR  41      LOST "MOL" DURING PACK ACKNOWLEDGE
1748  002162  064776                              EMT41
1749  002164  070526                              EHT1
1750  002166  070652                              EDT1
1751  002170  070742                              EFT1
1752
1753
1754                                    ;ERROR  42      UNSAFE ERROR DURING PACK ACKNOWLEDGE
1755  002172  065006                              EMT42
1756  002174  070526                              EHT1
1757  002176  070652                              EDT1
1758  002200  070742                              EFT1
1759
1760
1761                                    ;ERROR  43      "OPI" ERROR DURING PACK ACKNOWLEDGE
1762  002202  065020                              EMT43
1763  002204  070526                              EHT1
1764  002206  070652                              EDT1
1765  002210  070742                              EFT1
1766
1767
1768                                    ;ERROR  44      "RMR" ERROR DURING PACK ACKNOWLEDGE
1769  002212  065030                              EMT44
1770  002214  070526                              EHT1
1771  002216  070652                              EDT1
1772  002220  070742                              EFT1
1773
1774
1775                                    ;ERROR  45      "ILR" ERROR DURING PACK ACKNOWLEDGE
1776  002222  065040                              EMT45
1777  002224  070526                              EHT1
1778  002226  070652                              EDT1
1779  002230  070742                              EFT1
1780
1781
1782                                    ;ERROR  46      "ILF" ERROR DURING PACK ACKNOWLEDGE
1783  002232  065050                              EMT46
1784  002234  070526                              EHT1
1785  002236  070652                              EDT1
1786  002240  070742                              EFT1
1787
1788
1789                                    ;ERROR  47      COMPOSITE ERROR STATUS IS INCORRECT
1790  002242  065060                              EMT47
1791  002244  070526                              EHT1
1792  002246  070652                              EDT1
1793  002250  070742                              EFT1
1794
1795
1796                                    ;ERROR  50      PARITY ERROR WRITING REMOTE REGISTERS
1797  002252  065066                              EMT50
1798  002254  070526                              EHT1
1799  002256  070652                              EDT1
1800  002260  070742                              EFT1
1801
1802
```

```
1803                                    ;ERROR  51      INCORRECT IAE STATUS DURING SEEK COMMAND
1804  002262  065076                              EMT51
1805  002264  070526                              EHT1
1806  002266  070652                              EDT1
1807  002270  070742                              EFT1
1808
1809
1810                                    ;ERROR  52      OPI ERROR DURING SEEK - MEDIUM IS NOT ON LINE
1811  002272  065110                              EMT52
1812  002274  070526                              EHT1
1813  002276  070652                              EDT1
1814  002300  070742                              EFT1
1815
1816
1817                                    ;ERROR  53      OPI ERROR DURING SEEK - MEDIUM IS ON LINE, ASSUME
1818                                    ;               ON CYLINDER LATCH DIDN'T RESET
1819  002302  065126                              EMT53
1820  002304  070526                              EHT1
1821  002306  070652                              EDT1
1822  002310  070742                              EFT1
1823
1824
1825                                    ;ERROR  54      SEEK INCOMPLETE ERROR DURING SEEK COMMAND
1826  002312  065144                              EMT54
1827  002314  070526                              EHT1
1828  002316  070652                              EDT1
1829  002320  070742                              EFT1
1830
1831
1832                                    ;ERROR  55      DEVICE CHECK DURING SEEK COMMAND
1833  002322  065154                              EMT55
1834  002324  070526                              EHT1
1835  002326  070652                              EDT1
1836  002330  070742                              EFT1
1837
1838
1839                                    ;ERROR  56      PIP IS STILL SET AFTER SEEK - SKI IS RESET
1840  002332  065166                              EMT56
1841  002334  070526                              EHT1
1842  002336  070652                              EDT1
1843  002340  070742                              EFT1
1844
1845
1846                                    ;ERROR  57      ATA DID NOT SET DURING SEEK COMMAND
1847  002342  065204                              EMT57
1848  002344  070526                              EHT1
1849  002346  070652                              EDT1
1850  002350  070742                              EFT1
1851
1852
1853                                    ;ERROR  60      IVC ERROR DURING SEEK COMMAND - LOST
1854                                    ;               VOLUME VALID
1855  002352  065214                              EMT60
1856  002354  070526                              EHT1
1857  002356  070652                              EDT1
1858  002360  070742                              EFT1
```

```
                                ;ERROR  61      ERRONEOUS IVC ERROR DURING SEEK COMMAND -
                                ;               VOLUME VALID IS STIL SET
002362  065232                          EMT61
002364  070526                          EHT1
002366  070652                          EDT1
002370  070742                          EFT1


                                ;ERROR  62      MOL IS ZERO, BUT OPI WAS NOT
                                ;               REPORTED DURING SEEK COMMAND
002372  065252                          EMT62
002374  070526                          EHT1
002376  070652                          EDT1
002400  070742                          EFT1


                                ;ERROR  63      UNUSED
002402  000000                          0
002404  000000                          0
002406  000000                          0
002410  000000                          0


                                ;ERROR  64      DRIVE DID NOT DETECT "IVC" ERROR DURING SEEK
002412  065270                          EMT64
002414  070526                          EHT1
002416  070652                          EDT1
002420  070742                          EFT1


                                ;ERROR  65      DRIVE EXECUTED A SEEK WITH ERROR SET
002422  065310                          EMT65
002424  070526                          EHT1
002426  070652                          EDT1
002430  070742                          EFT1


                                ;ERROR  66      UNEXPECTED ERROR SET IN RMER1
002432  065330                          EMT66
002434  070526                          EHT1
002436  070652                          EDT1
002440  070742                          EFT1


                                ;ERROR  67      UNEXPECTED ERROR SET IN RMER2
002442  065342                          EMT67
002444  070526                          EHT1
002446  070652                          EDT1
002450  070742                          EFT1


                                ;ERROR  70      ERRONEOUS "IAE" ERROR DURING RECALIBRATE
002452  065354                          EMT70
002454  070526                          EHT1
```

```
1915  002456  070652                    EDT1
1916  002460  070742                    EFT1
1917
1918
1919                                ;ERROR  71      "ILF" ERROR DURING RECALIBRATE
1920  002462  065364                    EMT71
1921  002464  070526                    EHT1
1922  002466  070652                    EDT1
1923  002470  070742                    EFT1
1924
1925
1926                                ;ERROR  72      "OPI" ERROR DURING RECALIBRATE DUE TO "MOL" = 0
1927  002472  065374                    EMT72
1928  002474  070526                    EHT1
1929  002476  070652                    EDT1
1930  002500  070742                    EFT1
1931
1932
1933                                ;ERROR  73      "OPI" ERROR DURING RECALIBRATE BECAUSE ON
1934                                ;               CYLINDER DIDNT DROP
1935  002502  065412                    EMT73
1936  002504  070526                    EHT1
1937  002506  070652                    EDT1
1938  002510  070742                    EFT1
1939
1940
1941                                ;ERROR  74      "IVC" ERROR DURING RECALIBRATE - "VV" = 0
1942  002512  065430                    EMT74
1943  002514  070526                    EHT1
1944  002516  070652                    EDT1
1945  002520  070742                    EFT1
1946
1947
1948                                ;ERROR  75      ERRONEOUS "IVC" ERROR DURING RECALIBRATE - "VV" = 1
1949  002522  065440                    EMT75
1950  002524  070526                    EHT1
1951  002526  070652                    EDT1
1952  002530  070742                    EFT1
1953
1954
1955                                ;ERROR  76      "SKI" ERROR DURING RECALIBRATE
1956  002532  065460                    EMT76
1957  002534  070526                    EHT1
1958  002536  070652                    EDT1
1959  002540  070742                    EFT1
1960
1961
1962                                ;ERROR  77      "DVC" OCCURRED DURING RECALIBRATE
1963  002542  065470                    EMT77
1964  002544  070526                    EHT1
1965  002546  070652                    EDT1
1966  002550  070742                    EFT1
1967
19[illegible]
19[illegible]                       ;ERROR  100     LOST "MOL" DURING RECALIBRATE - "OPI" = 0
1970  002552  065502                    EMT100
```

```
 1971  002554  070526                    EHT1
 1972  002556  070652                    EDT1
 1973  002560  070742                    EFT1
 1974
 1975
 1976                              ;ERROR  101     LOST "VV" DURING RECALIBRATE - "IVC" = 0
 1977  002562  065520                    EMT101
 1978  002564  070526                    EHT1
 1979  002566  070652                    EDT1
 1980  002570  070742                    EFT1
 1981
 1982
 1983                              ;ERROR  102     "ATA" DID NOT SET DURING RECALIBRATE
 1984  002572  065536                    EMT102
 1985  002574  070526                    EHT1
 1986  002576  070652                    EDT1
 1987  002600  070742                    EFT1
 1988
 1989
 1990                              ;ERROR  103     "OM" DID NOT RESET DURING RECALIBRATE
 1991  002602  065546                    EMT103
 1992  002604  070526                    EHT1
 1993  002606  070652                    EDT1
 1994  002610  070742                    EFT1
 1995
 1996
 1997                              ;ERROR  104     "PIP" IS STIL SET AFTER RECALIBRATE
 1998  002612  065560                    EMT104
 1999  002614  070526                    EHT1
 2000  002616  070652                    EDT1
 2001  002620  070742                    EFT1
 2002
 2003
 2004                              ;ERROR  105     UNEXPECTED "ILR" ERROR DURING RECALIBRATE
 2005  002622  065576                    EMT105
 2006  002624  070526                    EHT1
 2007  002626  070652                    EDT1
 2008  002630  070742                    EFT1
 2009
 2010
 2011                              ;ERROR  106     UNEXPECTED "RMR" ERROR DURING RECALIBRATE
 2012  002632  065606                    EMT106
 2013  002634  070526                    EHT1
 2014  002636  070652                    EDT1
 2015  002640  070742                    EFT1
 2016
 2017
 2018                              ;ERROR  107     "UNS" ERROR DURING RECALIBRATE - AC POWER IS LOW
 2019  002642  065616                    EMT107
 2020  002644  070526                    EHT1
 2021  002646  070652                    EDT1
 2022  002650  070742                    EFT1
 2023
 2024
 2025                              ;ERROR  110     CANNOT ACCESS MASSBUS CONTROLLER VIA UNIBUS
 2026  002652  065636                    EMT110
```

```
 2027  002654  070550                    EHT110
 2028  002656  070670                    EDT110
 2029  002660  070760                    EFT110
 2030
 2031
 2032                              ;ERROR  111     NONEXISTENT DEVICE
 2033  002662  065650                    EMT111
 2034  002664  070554                    EHT111
 2035  002666  070672                    EDT111
 2036  002670  070762                    EFT111
 2037
 2038
 2039                              ;ERROR  112     DEVICE NOT AVAILABLE
 2040  002672  065656                    EMT112
 2041  002674  070554                    EHT111
 2042  002676  070672                    EDT111
 2043  002700  070762                    EFT111
 2044
 2045
 2046                              ;ERROR  113     BUS TIMEOUT-NED STATUS FAILURE
 2047  002702  065664                    EMT113
 2048  002704  000000                    0
 2049  002706  000000                    0
 2050  002710  000000                    0
 2051
 2052
 2053                              ;ERROR  114     DEVICE NOT AN RM03
 2054  002712  065700                    EMT114
 2055  002714  070560                    EHT114
 2056  002716  070674                    EDT114
 2057  002720  070764                    EFT114
 2058
 2059
 2060                              ;ERROR  115     RMCS1 NOT INITIALIZED BY UNIBUS
 2061  002722  065706                    EMT115
 2062  002724  070526                    EHT1
 2063  002726  070652                    EDT1
 2064  002730  070742                    EFT1
 2065
 2066
 2067                              ;ERROR  116     RMBA NOT INITIALIZED BY UNIBUS
 2068  002732  065716                    EMT116
 2069  002734  070526                    EHT1
 2070  002736  070652                    EDT1
 2071  002740  070742                    EFT1
 2072
 2073
 2074                              ;ERROR  117     RMCS2 NOT INITIALIZED BY UNIBUS
 2075  002742  065726                    EMT117
 2076  002744  070526                    EHT1
 2077  002746  070652                    EDT1
 2078  002750  070742                    EFT1
 2079
 2080
 2081                              ;ERROR  120     RMER1 NOT INITIALIZED BY UNIBUS
 2082  002752  065736                    EMT120
```

K04

ERROR POINTER TABLE

```
 2083  002754  070526                    EHT1
 2084  002756  070652                    EDT1
 2085  002760  070742                    EFT1
 2086
 2087
 2088                                    ;ERROR  121     RMAS NOT INITIALIZED BY UNIBUS
 2089  002762  065746                    EMT121
 2090  002764  070526                    EHT1
 2091  002766  070652                    EDT1
 2092  002770  070742                    EFT1
 2093
 2094
 2095                                    ;ERROR  122     RMMR1 NOT INITIALIZED BY UNIBUS
 2096  002772  065756                    EMT122
 2097  002774  070526                    EHT1
 2098  002776  070652                    EDT1
 2099  003000  070742                    EFT1
 2100
 2101
 2102                                    ;ERROR  123     RMDS NOT INITIALIZED BY UNIBUS
 2103  003002  065766                    EMT123
 2104  003004  070526                    EHT1
 2105  003006  070652                    EDT1
 2106  003010  070742                    EFT1
 2107
 2108
 2109                                    ;ERROR  124     RMEC2 NOT INITIALIZED BY UNIBUS
 2110  003012  065776                    EMT124
 2111  003014  070526                    EHT1
 2112  003016  070652                    EDT1
 2113  003020  070742                    EFT1
 2114
 2115
 2116                                    ;ERROR  125     RMMR2 NOT INITIALIZED BY UNIBUS
 2117  003022  066006                    EMT125
 2118  003024  070526                    EHT1
 2119  003026  070652                    EDT1
 2120  003030  070742                    EFT1
 2121
 2122
 2123                                    ;ERROR  126     RMCS1 NOT CLEARED BY CONTROLLER CLEAR
 2124  003032  066016                    EMT126
 2125  003034  070526                    EHT1
 2126  003036  070652                    EDT1
 2127  003040  070742                    EFT1
 2128
 2129
 2130                                    ;ERROR  127     RMBA NOT CLEARED BY CONTROLLER CLEAR
 2131  003042  066030                    EMT127
 2132  003044  070526                    EHT1
 2133  003046  070652                    EDT1
 2134  003050  070742                    EFT1
 2135
 2136
 2137                                    ;ERROR  130     RMCS2 NOT CLEARED BY CONTROLLER CLEAR
 2138  003052  066042                    EMT130
```

```
2139  003054  070526                    EHT1
2140  003056  070652                    EDT1
2141  003060  070742                    EFT1
2142
2143
2144                              ;ERROR  131     RMER1 NOT CLEARED BY CONTROLLER CLEAR
2145  003062  066054                    EMT131
2146  003064  070526                    EHT1
2147  003066  070652                    EDT1
2148  003070  070742                    EFT1
2149
2150
2151                              ;ERROR  132     RMAS NOT CLEARED BY CONTROLLER CLEAR
2152  003072  066066                    EMT132
2153  003074  070526                    EHT1
2154  003076  070652                    EDT1
2155  003100  070742                    EFT1
2156
2157
2158                              ;ERROR  133     RMMR1 NOT CLEARED BY CONTROLLER CLEAR
2159  003102  066100                    EMT133
2160  003104  070526                    EHT1
2161  003106  070652                    EDT1
2162  003110  070742                    EFT1
2163
2164
2165                              ;ERROR  134     RMDS NOT CLEARED BY CONTROLLER CLEAR
2166  003112  066112                    EMT134
2167  003114  070526                    EHT1
2168  003116  070652                    EDT1
2169  003120  070742                    EFT1
2170
2171
2172                              ;ERROR  135     RMEC2 NOT CLEARED BY CONTROLLER CLEAR
2173  003122  066124                    EMT135
2174  003124  070526                    EHT1
2175  003126  070652                    EDT1
2176  003130  070742                    EFT1
2177
2178
2179                              ;ERROR  136     RMMR2 NOT CLEARED BY CONTROLLER CLEAR
2180  003132  066136                    EMT136
2181  003134  070526                    EHT1
2182  003136  070652                    EDT1
2183  003140  070742                    EFT1
2184
2185
2186                              ;ERROR  137     RMCS1 NOT CLEARED BY ERROR CLEAR
2187  003142  066150                    EMT137
2188  003144  070526                    EHT1
2189  003146  070652                    EDT1
2190  003150  070742                    EFT1
2191
2192
2193                              ;ERROR  140     RMCS2 NOT CLEARED BY ERROR CLEAR
2194  003152  066160                    EMT140
```

ERROR POINTER TABLE

```
 2195  003154  070526                    EHT1
 2196  003156  070652                    EDT1
 2197  003160  070742                    EFT1
 2198
 2199
 2200                              ;ERROR  141     RMCS1 NOT CLEARED BY DRIVE CLEAR
 2201  003162  066170                    EMT141
 2202  003164  070526                    EHT1
 2203  003166  070652                    EDT1
 2204  003170  070742                    EFT1
 2205
 2206
 2207                              ;ERROR  142     RMDS NOT CLEARED BY DRIVE CLEAR
 2208  003172  066200                    EMT142
 2209  003174  070526                    EHT1
 2210  003176  070652                    EDT1
 2211  003200  070742                    EFT1
 2212
 2213
 2214                              ;ERROR  143     RMER1 NOT CLEARED BY DRIVE CLEAR
 2215  003202  066210                    EMT143
 2216  003204  070526                    EHT1
 2217  003206  070652                    EDT1
 2218  003210  070742                    EFT1
 2219
 2220
 2221                              ;ERROR  144     RMAS NOT CLEARED BY DRIVE CLEAR
 2222  003212  066220                    EMT144
 2223  003214  070526                    EHT1
 2224  003216  070652                    EDT1
 2225  003220  070742                    EFT1
 2226
 2227
 2228                              ;ERROR  145     RMMR1  NOT CLEARED BY DRIVE CLEAR
 2229  003222  066230                    EMT145
 2230  003224  070526                    EHT1
 2231  003226  070652                    EDT1
 2232  003230  070742                    EFT1
 2233
 2234
 2235                              ;ERROR  146     RMMR2 NOT CLEARED BY DRIVE CLEAR
 2236  003232  066240                    EMT146
 2237  003234  070526                    EHT1
 2238  003236  070652                    EDT1
 2239  003240  070742                    EFT1
 2240
 2241
 2242                              ;ERROR  147     RMER2 NOT CLEARED BY DRIVE CLEAR
 2243  003242  066250                    EMT147
 2244  003244  070526                    EHT1
 2245  003246  070652                    EDT1
 2246  003250  070742                    EFT1
 2247
 2248
 2249                              ;ERROR  150     RMEC2 NOT CLEARED BY DRIVE CLEAR
 2250  003252  066260                    EMT150
```

```
2251  003254  070526                    EHT1
2252  003256  070652                    EDT1
2253  003260  070742                    EFT1
2254
2255
2256                                    ;ERROR  151     MEDIUM NOT ON LINE
2257  003262  066270                    EMT151
2258  003264  070526                    EHT1
2259  003266  070652                    EDT1
2260  003270  070742                    EFT1
2261
2262
2263                                    ;ERROR  152     DRIVE FAULT
2264  003272  066302                    EMT152
2265  003274  070526                    EHT1
2266  003276  070652                    EDT1
2267  003300  070742                    EFT1
2268
2269
2270                                    ;ERROR  153     UNSAFE SHOULD BE SET BECAUSE DVC IS SET
2271  003302  066314                    EMT153
2272  003304  070526                    EHT1
2273  003306  070652                    EDT1
2274  003310  070742                    EFT1
2275
2276
2277                                    ;ERROR  154     UNSAFE SHOULD NOT BE SET, AC IS LOW
2278  003312  066332                    EMT154
2279  003314  070526                    EHT1
2280  003316  070652                    EDT1
2281  003320  070742                    EFT1
2282
2283
2284                                    ;ERROR  155     VOLUME VALID NOT SET BY PACK ACK
2285  003322  066350                    EMT155
2286  003324  070526                    EHT1
2287  003326  070652                    EDT1
2288  003330  070742                    EFT1
2289
2290
2291                                    ;ERROR  156     OFFSET MODE NOT SET BY OFFSET COMMAND
2292  003332  066362                    EMT156
2293  003334  070526                    EHT1
2294  003336  070652                    EDT1
2295  003340  070742                    EFT1
2296
2297
2298                                    ;ERROR  157     OFFSET MODE NOT RESET BY RTC COMMAND
2299  003342  066374                    EMT157
2300  003344  070526                    EHT1
2301  003346  070652                    EDT1
2302  003350  070742                    EFT1
2303
2304
2305                                    ;ERROR  160     RMOF NOT RESET BY RIP COMMAND
2306  003352  066406                    EMT160
```

ERROR POINTER TABLE

```
2307  003354  070526                        EHT1
2308  003356  070652                        EDT1
2309  003360  070742                        EFT1
2310
2311
2312                                ;ERROR  161     RMDA NOT RESET BY RIP COMMAND
2313  003362  066416                        EMT161
2314  003364  070526                        EHT1
2315  003366  070652                        EDT1
2316  003370  070742                        EFT1
2317
2318
2319                                ;ERROR  162     RMDC NOT RESET BY RIP COMMAND
2320  003372  066430                        EMT162
2321  003374  070526                        EHT1
2322  003376  070652                        EDT1
2323  003400  070742                        EFT1
2324
2325
2326                                ;ERROR 163      DATA WAS ECC CORRECTED BUT DOES NOT COMPARE WITH
2327                                ;               WRITE BUFFER
2328  003402  070322                        EMT336
2329  003404  070610                        EHT336
2330  003406  070706                        EDT336
2331  003410  070776                        EFT336
2332
2333
2334                                ;ERROR  164     OPI SHOULD NOT BE SET
2335  003412  066452                        EMT164
2336  003414  070526                        EHT1
2337  003416  070652                        EDT1
2338  003420  070742                        EFT1
2339
2340
2341                                ;ERROR  165     IVC SHOULD NOT BE SET
2342  003422  066460                        EMT165
2343  003424  070526                        EHT1
2344  003426  070652                        EDT1
2345  003430  070742                        EFT1
2346
2347
2348                                ;ERROR  166     IAE SHOULD NOT BE SET
2349  003432  066466                        EMT166
2350  003434  070526                        EHT1
2351  003436  070652                        EDT1
2352  003440  070742                        EFT1
2353
2354
2355                                ;ERROR  167     NEM SHOULD NOT BE SET
2356  003442  066474                        EMT167
2357  003444  070526                        EHT1
2358  003446  070652                        EDT1
2359  003450  070742                        EFT1
2360
2361
2362                                ;ERROR 170      UNUSED
```

ERROR POINTER TABLE

```
2363  003452  000000                    0
2364  003454  000000                    0
2365  003456  000000                    0
2366  003460  000000                    0
2367
2368
2369                          ;ERROR  171     "ATA" NOT SET DURING RETURN TO CENTERLINE
2370  003462  066512                    EMT171
2371  003464  070526                    EHT1
2372  003466  070652                    EDT1
2373  003470  070742                    EFT1
2374
2375
2376                          ;ERROR  172     "ATA" NOT SET BY OFFSET COMMAND
2377  003472  066522                    EMT172
2378  003474  070526                    EHT1
2379  003476  070652                    EDT1
2380  003500  070742                    EFT1
2381
2382
2383                          ;ERROR  173     RMER2 NOT INITIALIZED BY UNIBUS INIT
2384  003502  066532                    EMT173
2385  003504  070526                    EHT1
2386  003506  070652                    EDT1
2387  003510  070742                    EFT1
2388
2389
2390                          ;ERROR  174     RMER2 NOT INITIALIZED BY CONTROLLER CLEAR
2391  003512  066542                    EMT174
2392  003514  070526                    EHT1
2393  003516  070652                    EDT1
2394  003520  070742                    EFT1
2395
2396
2397                          ;ERROR  175     SELECTED DEVICE IS IN WRITE PROTECT
2398  003522  066554                    EMT175
2399  003524  070526                    EHT1
2400  003526  070652                    EDT1
2401  003530  070742                    EFT1
2402
2403
2404                          ;ERROR  176     CANNOT SET DIAGNOSTIC MODE
2405  003532  066562                    EMT176
2406  003534  070526                    EHT1
2407  003536  070652                    EDT1
2408  003540  070742                    EFT1
2409
2410
2411                          ;ERROR  177     INCORRECT "MOL" STATUS DURING DIAGNOSTIC MODE
2412  003542  066570                    EMT177
2413  003544  070526                    EHT1
2414  003546  070652                    EDT1
2415  003550  070742                    EFT1
2416
2417
2418                          ;ERROR  200     INCORRECT "PIP" STATUS DURING DIAGNOSTIC MODE
```

ERROR POINTER TABLE

```
2419  003552  066602                  EMT200
2420  003554  070526                  EHT1
2421  003556  070652                  EDT1
2422  003560  070742                  EFT1
2423
2424
2425                          ;ERROR  201     INCORRECT "WRL" STATUS DURING DIAGNOSTIC MODE
2426  003562  066614                  EMT201
2427  003564  070526                  EHT1
2428  003566  070652                  EDT1
2429  003570  070742                  EFT1
2430
2431
2432                          ;ERROR  202     INCORRECT "SKI" STATUS DURING DIAGNOSTIC MODE
2433  003572  066626                  EMT202
2434  003574  070526                  EHT1
2435  003576  070652                  EDT1
2436  003600  070742                  EFT1
2437
2438
2439                          ;ERROR  203     INCORRECT "DVC" STATUS DURING DIAGNOSTIC MODE
2440  003602  066640                  EMT203
2441  003604  070526                  EHT1
2442  003606  070652                  EDT1
2443  003610  070742                  EFT1
2444
2445
2446                          ;ERROR  204     "VV" WAS NOT RESET BY MAINTENANCE UNIT READY
2447  003612  066652                  EMT204
2448  003614  070526                  EHT1
2449  003616  070652                  EDT1
2450  003620  070742                  EFT1
2451
2452
2453                          ;ERROR  205     SELECTED DEVICE HAS A PERSISTENT "SKI" ERROR
2454  003622  066670                  EMT205
2455  003624  070526                  EHT1
2456  003626  070652                  EDT1
2457  003630  070742                  EFT1
2458
2459
2460                          ;ERROR  206     "LBC" DID NOT SET DURING DIAGNOSTIC MODE
2461  003632  066700                  EMT206
2462  003634  070526                  EHT1
2463  003636  070652                  EDT1
2464  003640  070742                  EFT1
2465
2466
2467                          ;ERROR  207     UNEXPECTED LOSS OF "MOL" - MEDIUM IS OFF LINE
2468  003642  066710                  EMT207
2469  003644  070526                  EHT1
2470  003646  070652                  EDT1
2471  003650  070742                  EFT1
2472
2473
2474                          ;ERROR  210     UNEXPECTED LOSS OF VOLUME VALID - "VV" = 0
```

```
2475  003652  066722                    EMT210
2476  003654  070526                    EHT1
2477  003656  070652                    EDT1
2478  003660  070742                    EFT1
2479
2480
2481                              ;ERROR  211     UNEXPECTED MECHANICAL MOTION - "PIP" = 1
2482  003662  066730                    EMT211
2483  003664  070526                    EHT1
2484  003666  070652                    EDT1
2485  003670  070742                    EFT1
2486
2487
2488                              ;ERROR  212     UNEXPECTED DEVICE FAULT - "DVC" = 1
2489  003672  066744                    EMT212
2490  003674  070526                    EHT1
2491  003676  070652                    EDT1
2492  003700  070742                    EFT1
2493
2494
2495                              ;ERROR  213     UNEXPECTED SEEK INCOMPLETE ERROR - "SKI" = 1
2496  003702  066760                    EMT213
2497  003704  070526                    EHT1
2498  003706  070652                    EDT1
2499  003710  070742                    EFT1
2500
2501
2502                              ;ERROR  214     DRIVE EXECUTED A RECALIBRATE WITH ERROR SET
2503  003712  066770                    EMT214
2504  003714  070540                    EHT2
2505  003716  070662                    EDT2
2506  003720  070752                    EFT2
2507
2508
2509                              ;ERROR  215     DRIVE DID NOT DETECT "IVC" ERROR DURING RECALIBRATE
2510  003722  067010                    EMT215
2511  003724  070540                    EHT2
2512  003726  070662                    EDT2
2513  003730  070752                    EFT2
2514
2515
2516                              ;ERROR  216     INCORRECT "IVC" STATUS
2517  003732  067022                    EMT216
2518  003734  070526                    EHT1
2519  003736  070652                    EDT1
2520  003740  070742                    EFT1
2521
2522
2523                              ;ERROR  217     INCORRECT "IAE" STATUS
2524  003742  067032                    EMT217
2525  003744  070526                    EHT1
2526  003746  070652                    EDT1
2527  003750  070742                    EFT1
2528
2529
2530                              ;ERROR 220      INCORRECT "WLE" STATUS
```

ERROR POINTER TABLE

```
2531  003752  067042                    EMT220
2532  003754  070526                    EHT1
2533  003756  070652                    EDT1
2534  003760  070742                    EFT1
2535
2536
2537                              ;ERROR  221     INCORRECT "OPI" STATUS
2538  003762  067052                    EMT221
2539  003764  070526                    EHT1
2540  003766  070652                    EDT1
2541  003770  070742                    EFT1
2542
2543
2544                              ;ERROR  222     RM03 DID NOT DETECT RMR ERROR
2545  003772  067062                    EMT222
2546  003774  070526                    EHT1
2547  003776  070652                    EDT1
2548  004000  070742                    EFT1
2549
2550
2551                              ;ERROR  223     RM03 DID NOT DETECT PARITY ERROR ON MASSBUS CONTROL BUS
2552  004002  067072                    EMT223
2553  004004  070564                    EHT223
2554  004006  070676                    EDT223
2555  004010  070766                    EFT223
2556
2557
2558                              ;ERROR  224     UNUSED
2559  004012  000000                    0
2560  004014  000000                    0
2561  004016  000000                    0
2562  004020  000000                    0
2563
2564
2565                              ;ERROR  225     UNUSED
2566  004022  000000                    0
2567  004024  000000                    0
2568  004026  000000                    0
2569  004030  000000                    0
2570
2571
2572                              ;ERROR  226     UNUSED
2573  004032  000000                    0
2574  004034  000000                    0
2575  004036  000000                    0
2576  004040  000000                    0
2577
2578
2579                              ;ERROR  227     UNUSED
2580  004042  000000                    0
2581  004044  000000                    0
2582  004046  000000                    0
2583  004050  000000                    0
2584
2585
2586                              ;ERROR  230     UNUSED
```

```
2587  004052  000000                    0
2588  004054  000000                    0
2589  004056  000000                    0
2590  004060  000000                    0
2591
2592
2593                              ;ERROR  231     UNUSED
2594  004062  000000                    0
2595  004064  000000                    0
2596  004066  000000                    0
2597  004070  000000                    0
2598
2599
2600                              ;ERROR  232     UNUSED
2601  004072  000000                    0
2602  004074  000000                    0
2603  004076  000000                    0
2604  004100  000000                    0
2605
2606
2607                              ;ERROR  233     UNUSED
2608  004102  000000                    0
2609  004104  000000                    0
2610  004106  000000                    0
2611  004110  000000                    0
2612
2613
2614                              ;ERROR  234     UNUSED
2615  004112  000000                    0
2616  004114  000000                    0
2617  004116  000000                    0
2618  004120  000000                    0
2619
2620
2621                              ;ERROR  235     UNUSED
2622  004122  000000                    0
2623  004124  000000                    0
2624  004126  000000                    0
2625  004130  000000                    0
2626
2627
2628                              ;ERROR  236     UNUSED
2629  004132  000000                    0
2630  004134  000000                    0
2631  004136  000000                    0
2632  004140  000000                    0
2633
2634
2635                              ;ERROR  237     UNUSED
2636  004142  000000                    0
2637  004144  000000                    0
2638  004146  000000                    0
2639  004150  000000                    0
2640
2641
2642                              ;ERROR  240     UNUSED
```

```
2643  004152  000000                    0
2644  004154  000000                    0
2645  004156  000000                    0
2646  004160  000000                    0
2647
2648
2649                                ;ERROR  241     UNUSED
2650  004162  000000                    0
2651  004164  000000                    0
2652  004166  000000                    0
2653  004170  000000                    0
2654
2655
2656                                ;ERROR  242     UNUSED
2657  004172  000000                    0
2658  004174  000000                    0
2659  004176  000000                    0
2660  004200  000000                    0
2661
2662
2663                                ;ERROR  243     UNUSED
2664  004202  000000                    0
2665  004204  000000                    0
2666  004206  000000                    0
2667  004210  000000                    0
2668
2669
2670                                ;ERROR  244     UNUSED
2671  004212  000000                    0
2672  004214  000000                    0
2673  004216  000000                    0
2674  004220  000000                    0
2675
2676
2677                                ;ERROR  245     UNUSED
2678  004222  000000                    0
2679  004224  000000                    0
2680  004226  000000                    0
2681  004230  000000                    0
2682
2683
2684                                ;ERROR  246     "ATA" NOT RESET BY GO WHEN "ERR" = 0
2685  004232  067146                    EMT246
2686  004234  070526                    EHT1
2687  004236  070652                    EDT1
2688  004240  070742                    EFT1
2689
2690
2691                                ;ERROR  247     "ATA" NOT RESET BY WRITING RMAS
2692  004242  067156                    EMT247
2693  004244  070526                    EHT1
2694  004246  070652                    EDT1
2695  004250  070742                    EFT1
2696
2697
2698                                ;ERROR  250     "ATA" WAS RESET BY GO WHEN "ERR" = 1
```

ERROR POINTER TABLE

```
2699  004252  067170                    EMT250
2700  004254  070526                    EHT1
2701  004256  070652                    EDT1
2702  004260  070742                    EFT1
2703
2704
2705                          ;ERROR  251     PROGRAM INTERRUPT WAS NOT GENERATED
2706  004262  067204                    EMT251
2707  004264  070540                    EHT2
2708  004266  070662                    EDT2
2709  004270  070752                    EFT2
2710
2711
2712                          ;ERROR  252     PROGRAM INTERRUPT SHOULD NOT HAVE BEEN GENERATED
2713  004272  067212                    EMT252
2714  004274  070540                    EHT2
2715  004276  070662                    EDT2
2716  004300  070752                    EFT2
2717
2718
2719                          ;ERROR  253     OFFSET MODE WAS NOT RESET BY WRITING RMDC
2720  004302  067220                    EMT253
2721  004304  070526                    EHT1
2722  004306  070652                    EDT1
2723  004310  070742                    EFT1
2724
2725
2726                          ;ERROR  254     INCORRECT "ILF" STATUS
2727  004312  067236                    EMT254
2728  004314  070526                    EHT1
2729  004316  070652                    EDT1
2730  004320  070742                    EFT1
2731
2732
2733                          ;ERROR  255     INCORRECT "ATA" STATUS
2734  004322  067246                    EMT255
2735  004324  070526                    EHT1
2736  004326  070652                    EDT1
2737  004330  070742                    EFT1
2738
2739
2740                          ;ERROR  256     INCORRECT "ILR" STATUS
2741  004332  067256                    EMT256
2742  004334  070576                    EHT256
2743  004336  070676                    EDT223
2744  004340  070766                    EFT223
2745
2746
2747                          ;ERROR  257     INVALID IAE STATUS DURING SEARCH COMMAND
2748  004342  067266                    EMT257
2749  004344  070526                    EHT1
2750  004346  070652                    EDT1
2751  004350  070742                    EFT1
2752
2753
2754                          ;ERROR  260     "IVC" WAS NOT DETECTED DURING SEARCH COMMAND
```

ERROR POINTER TABLE

```
2755  004352  067300                    EMT260
2756  004354  070526                    EHT1
2757  004356  070652                    EDT1
2758  004360  070742                    EFT1
2759
2760
2761                          ;ERROR  261     DRIVE EXECUTED SEARCH WITH ERROR SET
2762  004362  067312                    EMT261
2763  004364  070526                    EHT1
2764  004366  070652                    EDT1
2765  004370  070742                    EFT1
2766
2767
2768                          ;ERROR  262     "LBC" ERROR NOT SET DURING DIAGNOSTIC MODE
2769  004372  067332                    EMT262
2770  004374  070526                    EHT1
2771  004376  070652                    EDT1
2772  004400  070742                    EFT1
2773
2774
2775                          ;ERROR  263     "SKI" ERROR DURING SEARCH COMMAND
2776  004402  067342                    EMT263
2777  004404  070526                    EHT1
2778  004406  070652                    EDT1
2779  004410  070742                    EFT1
2780
2781
2782                          ;ERROR  264     "IVC" ERROR DURING SEARCH - LOST VOLUME VALID
2783  004412  067352                    EMT264
2784  004414  070526                    EHT1
2785  004416  070652                    EDT1
2786  004420  070742                    EFT1
2787
2788
2789                          ;ERROR  265     ERRONEOUS IVC ERROR DURING SEARCH - VOLUME IS VALID
2790  004422  067372                    EMT265
2791  004424  070526                    EHT1
2792  004426  070652                    EDT1
2793  004430  070742                    EFT1
2794
2795
2796                          ;ERROR  266     DEVICE FAULT (DVC) DURING SEARCH
2797  004432  067412                    EMT266
2798  004434  070526                    EHT1
2799  004436  070652                    EDT1
2800  004440  070742                    EFT1
2801
2802
2803                          ;ERROR  267     SKI SHOULD HAVE BEEN SET BECAUSE CYLINDER
2804                          ;               ADDRESS IS TOO LARGE
2805  004442  067424                    EMT267
2806  004444  070526                    EHT1
2807  004446  070652                    EDT1
2808  004450  070742                    EFT1
2809
2810
```

```
2811
2812  004452  067442                ;ERROR  270     OPI ERROR DURING SEARCH BECAUSE MOL = 0
2813  004454  070526                        EMT270
2814  004456  070652                        EHT1
2815  004460  070742                        EDT1
2816                                  EFT1
2817
2818                                ;ERROR  271     OPI ERROR DURING SEARCH BECAUSE ON CYLINDER
2819                                ;               DIDN'T DROP
2820  004462  067456                        EMT271
2821  004464  070526                        EHT1
2822  004466  070652                        EDT1
2823  004470  070742                        EFT1
2824
2825
2826                                ;ERROR  272     LOST MOL DURING SEARCH, OPI IS NOT SET
2827  004472  067474                        EMT272
2828  004474  070526                        EHT1
2829  004476  070652                        EDT1
2830  004500  070742                        EFT1
2831
2832
2833                                ;ERROR  273     PIP STIL SET AFTER SEARCH
2834  004502  067512                        EMT273
2835  004504  070526                        EHT1
2836  004506  070652                        EDT1
2837  004510  070742                        EFT1
2838
2839
2840                                ;ERROR  274     PARITY ERROR OCCURRED WHILE WRITING REMOTE
2841                                ;               REGISTERS BUT MXF DID NOT SET
2842  004512  067530                        EMT274
2843  004514  070526                        EHT1
2844  004516  070652                        EDT1
2845  004520  070742                        EFT1
2846
2847
2848                                ;ERROR  275     MXF ERROR - COMPOSITE ERROR OCCURRED BEFORE DATA
2849                                ;               COMMAND STARTED
2850  004522  067546                        EMT275
2851  004524  070526                        EHT1
2852  004526  070652                        EDT1
2853  004530  070742                        EFT1
2854
2855
2856                                ;ERROR  276     "OPI" ERROR DURING DATA TRANSFER BECAUSE "MOL" WAS
2857                                ;               ZERO
2858  004532  067560                        EMT276
2859  004534  070526                        EHT1
2860  004536  070652                        EDT1
2861  004540  070742                        EFT1
2862
2863
2864                                ;ERROR  277     "OPI" ERROR DURING DATA TRANSFER BECAUSE 1) ON
2865                                ;               CYLINDER DIDN'T DROP OR 2) SEARCH TIMED OUT OR
2866                                ;               3) RUN TIMED OUT
```

ERROR POINTER TABLE

```
2867  004542  067574                        EMT277
2868  004544  070526                        EHT1
2869  004546  070652                        EDT1
2870  004550  070742                        EFT1
2871
2872
2873                          ;ERROR  300     "IVC" ERROR DURING DATA TRANSFER BECAUSE VOLUME
2874                          ;               WAS NOT VALID
2875  004552  067612                        EMT300
2876  004554  070526                        EHT1
2877  004556  070652                        EDT1
2878  004560  070742                        EFT1
2879
2880
2881                          ;ERROR  301     ERRONEOUS "IVC" ERROR DURING DATA TRANSFER - VOLUME
2882                          ;               IS VALID
2883  004562  067632                        EMT301
2884  004564  070526                        EHT1
2885  004566  070652                        EDT1
2886  004570  070742                        EFT1
2887
2888
2889                          ;ERROR  302     "ILR" ERROR DURING DATA TRANSFER
2890  004572  067654                        EMT302
2891  004574  070526                        EHT1
2892  004576  070652                        EDT1
2893  004600  070742                        EFT1
2894
2895
2896                          ;ERROR  303     "ILF" ERROR DURING DATA TRANSFER
2897  004602  067666                        EMT303
2898  004604  070526                        EHT1
2899  004606  070652                        EDT1
2900  004610  070742                        EFT1
2901
2902
2903                          ;ERROR  304     "RMR" ERROR DURING DATA TRANSFER
2904  004612  067700                        EMT304
2905  004614  070526                        EHT1
2906  004616  070652                        EDT1
2907  004620  070742                        EFT1
2908
2909
2910                          ;ERROR  305     INCORRECT "IAE" STATUS DURING DATA TRANSFER
2911  004622  067712                        EMT305
2912  004624  070526                        EHT1
2913  004626  070652                        EDT1
2914  004630  070742                        EFT1
2915
2916
2917                          ;ERROR  306     "SKI" ERROR DURING DATA TRANSFER
2918  004632  067724                        EMT306
2919  004634  070526                        EHT1
2920  004636  070652                        EDT1
2921  004640  070742                        EFT1
2922
```

```
2923
2924                                    ;ERROR  307     DRIVE DID NOT DETECT SKI ERROR DUE TO CYLINDER
2925  004642  067734                            EMT307
2926  004644  070526                            EHT1
2927  004646  070652                            EDT1
2928  004650  070742                            EFT1
2929
2930
2931                                    ;ERROR  310     DEVICE FAULT DURING DATA TRANSFER
2932  004652  067754                            EMT310
2933  004654  070526                            EHT1
2934  004656  070652                            EDT1
2935  004660  070742                            EFT1
2936
2937
2938                                    ;ERROR  311     LOSS OF BIT CLOCK DURING DATA TRANSFER
2939  004662  067766                            EMT311
2940  004664  070526                            EHT1
2941  004666  070652                            EDT1
2942  004670  070742                            EFT1
2943
2944
2945                                    ;ERROR  312     LOSS OF SYSTEM CLOCK DURING DATA TRANSFER
2946  004672  070000                            EMT312
2947  004674  070526                            EHT1
2948  004676  070652                            EDT1
2949  004700  070742                            EFT1
2950
2951
2952                                    ;ERROR  313     UNSAFE ERROR DURING DATA TRANSFER (DVC = 0)
2953  004702  070012                            EMT313
2954  004704  070526                            EHT1
2955  004706  070652                            EDT1
2956  004710  070742                            EFT1
2957
2958
2959                                    ;ERROR  314     DRIVE TIMING ERROR DURING DATA TRANSFER
2960  004712  070032                            EMT314
2961  004714  070526                            EHT1
2962  004716  070652                            EDT1
2963  004720  070742                            EFT1
2964
2965
2966                                    ;ERROR  315     WRITE LOCK ERROR
2967  004722  070044                            EMT315
2968  004724  070526                            EHT1
2969  004726  070652                            EDT1
2970  004730  070742                            EFT1
2971
2972
2973                                    ;ERROR  316     ERRONEOUS WRITE LOCK ERROR
2974  004732  070056                            EMT316
2975  004734  070526                            EHT1
2976  004736  070652                            EDT1
2977  004740  070742                            EFT1
2978
```

```
2979
2980
2981  004742  070070                ;ERROR  317     HEADER CRC ERROR DURING DATA TRANSFER
2982  004744  070526                        EMT317
2983  004746  070652                        EHT1
2984  004750  070742                        EDT1
2985                                        EFT1
2986
2987                                ;ERROR  320     FORMAT ERROR DURING DATA TRANSFER
2988  004752  070100                        EMT320
2989  004754  070526                        EHT1
2990  004756  070652                        EDT1
2991  004760  070742                        EFT1
2992
2993
2994                                ;ERROR  321     HEADER COMPARE ERROR DURING DATA TRANSFER
2995  004762  070110                        EMT321
2996  004764  070526                        EHT1
2997  004766  070652                        EDT1
2998  004770  070742                        EFT1
2999
3000
3001                                ;ERROR  322     HEADER ERRORS SHOULD NOT BE SET
3002  004772  070120                        EMT322
3003  004774  070526                        EHT1
3004  004776  070652                        EDT1
3005  005000  070742                        EFT1
3006
3007
3008                                ;ERROR  323     DATA CHECK ERROR DURING DATA TRANSFER
3009  005002  070126                        EMT323
3010  005004  070526                        EHT1
3011  005006  070652                        EDT1
3012  005010  070742                        EFT1
3013
3014
3015                                ;ERROR  324     CORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
3016  005012  070136                        EMT324
3017  005014  070526                        EHT1
3018  005016  070652                        EDT1
3019  005020  070742                        EFT1
3020
3021
3022                                ;ERROR  325     UNCORRECTABLE DATA CHECK ERROR DURING DATA TRANSFER
3023  005022  070150                        EMT325
3024  005024  070526                        EHT1
3025  005026  070652                        EDT1
3026  005030  070742                        EFT1
3027
3028
3029                                ;ERROR  326     DATA PARITY ERROR DURING READ COMMMAND
3030  005032  070162                        EMT326
3031  005034  070526                        EHT1
3032  005036  070652                        EDT1
3033  005040  070742                        EFT1
3034
```

```
3035
3036                                    ;ERROR  327     OFFSET MODE NOT RESET BY WRITE COMMAND
3037  005042  070200                              EMT327
3038  005044  070526                              EHT1
3039  005046  070652                              EDT1
3040  005050  070742                              EFT1
3041
3042
3043                                    ;ERROR  330     DATA PARITY ERROR DURING WRITE COMMAND
3044  005052  070212                              EMT330
3045  005054  070526                              EHT1
3046  005056  070652                              EDT1
3047  005060  070742                              EFT1
3048
3049
3050                                    ;ERROR  331     WRITE CLOCK FALURE DURING WRITE COMMAND
3051  005062  070222                              EMT331
3052  005064  070526                              EHT1
3053  005066  070652                              EDT1
3054  005070  070742                              EFT1
3055
3056
3057                                    ;ERROR  332     DATA LATE ERROR DURING DATA TRANSFER
3058  005072  070234                              EMT332
3059  005074  070526                              EHT1
3060  005076  070652                              EDT1
3061  005100  070742                              EFT1
3062
3063
3064                                    ;ERROR  333     PIP STIL SET AFTER DATA TRANSFER - SKI = 0
3065  005102  070246                              EMT333
3066  005104  070526                              EHT1
3067  005106  070652                              EDT1
3068  005110  070742                              EFT1
3069
3070
3071                                    ;ERROR  334     LOST MOL DURING DATA TRANSFER - OPI = 0
3072  005112  070264                              EMT334
3073  005114  070526                              EHT1
3074  005116  070652                              EDT1
3075  005120  070742                              EFT1
3076
3077
3078                                    ;ERROR  335     LOST VOLUME VALID DURING DATA TRANSFER - IVC  = 0
3079  005122  070302                              EMT335
3080  005124  070526                              EHT1
3081  005126  070652                              EDT1
3082  005130  070742                              EFT1
3083
3084
3085                                    ;ERROR  336     DATA READ DOES NOT COMPARE WITH DATA WRITTEN
3086  005132  070322                              EMT336
3087  005134  070610                              EHT336
3088  005136  070706                              EDT336
3089  005140  070776                              EFT336
3090
```

```
3091
3092                                    ;ERROR  337     WRITE CHECK ERROR NOT DETECTED
3093  005142  070332                              EMT337
3094  005144  070622                              EHT337
3095  005146  070716                              EDT337
3096  005150  071006                              EFT337
3097
3098
3099                                    ;ERROR  340     WRITE CHECK ERROR AT UNEXPECTED ADDRESS
3100  005152  070342                              EMT340
3101  005154  070610                              EHT336
3102  005156  070706                              EDT336
3103  005160  070776                              EFT336
3104
3105
3106                                    ;ERROR  341     INCORRECT DATA DURING WRITE CHECK ERROR
3107  005162  070354                              EMT341
3108  005164  070610                              EHT336
3109  005166  070706                              EDT336
3110  005170  070776                              EFT336
3111
3112
3113                                    ;ERROR  342     "IVC" ERROR NOT DETECTED DURING DATA TRANSFER
3114  005172  070362                              EMT342
3115  005174  070526                              EHT1
3116  005176  070652                              EDT1
3117  005200  070742                              EFT1
3118
3119
3120                                    ;ERROR  343     "FER" NOT DETECTED DURING DATA TRANSFER
3121  005202  070374                              EMT343
3122  005204  070526                              EHT1
3123  005206  070652                              EDT1
3124  005210  070742                              EFT1
3125
3126
3127                                    ;ERROR  344     "HCE" NOT DETECTED DURING DATA TRANSFER
3128  005212  070406                              EMT344
3129  005214  070634                              EHT344
3130  005216  070726                              EDT344
3131  005220  071016                              EFT344
3132
3133
3134                                    ;ERROR  345     "BSE" NOT DETECTED DURING DATA TRANSFER
3135  005222  070420                              EMT345
3136  005224  070526                              EHT1
3137  005226  070652                              EDT1
3138  005230  070742                              EFT1
3139
3140
3141                                    ;ERROR  346     HEADER ERROR WAS DETECTED W/ HCI SET
3142  005232  070430                              EMT346
3143  005234  070526                              EHT1
3144  005236  070652                              EDT1
3145  005240  070742                              EFT1
3146
```

```
3147
3148                                    ;ERROR  347     DATA TRANSFER NOT ABORTED W/ COMP ERROR SET
3149  005242  070444                              EMT347
3150  005244  070526                              EHT1
3151  005246  070652                              EDT1
3152  005250  070742                              EFT1
3153
3154
3155                                    ;ERROR  350     LOST VOLUME VALID DURING SEARCH - "IVC" = 0
3156  005252  070456                              EMT350
3157  005254  070526                              EHT1
3158  005256  070652                              EDT1
3159  005260  070742                              EFT1
3160
3161
3162                                    ;ERROR  351     "ATA" DID NOT SET DURING SEARCH
3163  005262  070474                              EMT351
3164  005264  070526                              EHT1
3165  005266  070652                              EDT1
3166  005270  070742                              EFT1
3167
3168
3169                                    ;ERROR  352     PROGRAM TIMEOUT WHILE TESTING RMLA
3170  005272  070504                              EMT352
3171  005274  000000                              0
3172  005276  000000                              0
3173  005300  000000                              0
3174
3175
3176                                    ;ERROR  353     LOOK AHEAD TEST FAILS
3177  005302  070510                              EMT353
3178  005304  070646                              EHT353
3179  005306  070740                              EDT353
3180  005310  071026                              EFT353
3181
3182
3183                                    ;ERROR  354     BSE SHOULD NOT BE SET
3184  005312  070520                              EMT354
3185  005314  070526                              EHT1
3186  005316  070652                              EDT1
3187  005320  070742                              EFT1
3188
3189
3190                                    ;PUT ERROR TABLE HERE
3191                                    .SBTTL  ERROR TABLE USAGE
3192
3193                                    ;THE ERROR TABLE ABOVE CONSISTS OF FOUR WORD ENTRIES FOR EACH ERROR
3194                                    ;NUMBER, I.E.,
3195                                    ;
3196                                    ;       EMT - ERROR MESSAGE TABLE ADDRESS
3197                                    ;       EHT - ERROR HEADER TABLE ADDRESS
3198                                    ;       EDT - ERROR DATA TABLE ADDRESS
3199                                    ;       EFT - ERROR FORMAT TABLE ADDRESS
3200                                    ;
3201                                    ;THE EMT ENTRY IS THE ADDRESS OF THE TABLE OF ERROR MESSAGE STRINGS
3202                                    ;FOR THE PARTICULAR ERROR.  EACH ERROR MESSAGE TABLE LISTS THE ADDRESS
```

```
3203                    ;OF ONE OR MORE ERROR MESSAGE STRINGS WHICH ARE TO BE FORMATTED AND
3204                    ;TYPED BY THE ERROR TYPE SUBROUTINE.  IF THE EMT ENTRY IS ZERO, THERE IS
3205                    ;NO MESSAGE TO BE TYPED FOR THE ERROR.
3206                    ;
3207                    ;SIMILARLY, THE EHT, EDT, AND EFT ENTRIES ARE ADDRESSES OF TABLES
3208                    ;OF HEADER, DATA AND FORMAT INFORMATION FOR A GIVEN ERROR.  EACH ENTRY
3209                    ;IN THE ERROR HEADER TABLE MAY OR MAY NOT HAVE AN ASSOCIATED LINE OF
3210                    ;DATA, HOWEVER, EACH DATA LINE MUST HAVE AN ASSOCIATED FORMAT AND
3211                    ;HEADER. THAT IS, A HEADER LINE MAY BE PRINTED WITHOUT ANY DATA,
3212                    ;BUT A DATA LINE IS NOT PRINTED WITHOUT A HEADER, AND EACH DATA LINE
3213                    ;MUST ALSO HAVE A FORMAT.
3214                    ;
3215                    ;IN SUMMARY,
3216                    ;
3217                    ;       EACH NONZERO ENTRY IS THE ADDRESS OF A TABLE,
3218                    ;       EACH TABLE IS A LIST OF ADDRESSES WHICH DEFINES THE LOCATIONS
3219                    ;OF MESSAGE STRINGS, HEADERS, DATA OR FORMAT.
3220                    ;
3221
```

```
                                        .SBTTL  START OF PROGRAM


005322                                  START:

                                        ;CLEAR AND SETUP FOR TEST EXECUTION
005322  000240                                 NOP
005324  000005                                 RESET                   ;INITIALIZE THE SYSTEM
005326  013746  000300                         MOV     PR6,-(SP)       ;;PUT NEW PS ON STACK
005332  012746  005340                         MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
005336  000002                                 RTI                     ;;POP NEW PC AND PS
005340                                  64$:

                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
005340  012706  001114                         MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
005344  005026                                 CLR     (R6)+           ;;CLEAR MEMORY LOCATION
005346  022706  001154                         CMP     #$WR,R6 ;;DONE?
005352  001374                                 BNE     .-6             ;;LOOP BACK IF NO
005354  012706  001100                         MOV     #STACK,SP       ;;SETUP THE STACK POINTER

                                        ;;INITIALIZE A FEW VECTORS
005360  012737  060032  000020                 MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
005366  012737  000340  000022                 MOV     #340,@#IOTVEC+2 ;;LEVEL 7
005374  012737  060442  000030                 MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
005402  012737  000340  000032                 MOV     #340,@#EMTVEC+2 ;;LEVEL 7
005410  012737  062202  000034                 MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
005416  012737  000340  000036                 MOV     #340,@#TRAPVEC+2;LEVEL 7
005424  012737  062310  000024                 MOV     #$PWRON,@#PWRVEC ;;POWER FAILURE VECTOR
005432  012737  000340  000026                 MOV     #340,@#PWRVEC+2 ;;LEVEL 7
005440  013737  032316  032310                 MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
005446  005037  001206                         CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
005452  005037  001210                         CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
005456  112737  000001  001131                 MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
005464  012737  005464  001122                 MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
005472  012737  005472  001124                 MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
005500  013746  000004                         MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
005504  012737  005540  000004                 MOV     #65$,@#ERRVEC   ;;SET UP ERROR VECTOR
005512  012737  177570  001154                 MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
005520  012737  177570  001156                 MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
005526  022777  177777  173420                 CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
005534  001012                                 BNE     67$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                       ;;AND  THE HARDWARE SWR IS NOT = -1
005536  000403                                 BR      66$             ;;BRANCH IF NO TIMEOUT
005540  012716  005546          65$:           MOV     #66$,(SP)       ;;SET UP FOR TRAP RETURN
005544  000002                                 RTI
005546  012737  000176  001154  66$:           MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
005554  012737  000174  001156                 MOV     #DISPREG,DISPLAY
005562  012637  000004          67$:           MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

005566  005037  001230                         CLR     $PASS           ;;CLEAR PASS COUNT
005572  132737  000200  001243                 BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
005600  001403                                 BEQ     68$             ;;YES,USE NON-APT SWITCH
005602  012737  001244  001154                 MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
005610                                  68$:
```

```
                                        .SBTTL  TYPE PROGRAM NAME
                                        ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
005610  005227  177777                          INC     #-1             ;;FIRST TIME?
005614  001060                                  BNE     69$             ;;BRANCH IF NO
005616  022737  032452  000042                  CMP     #SENDAD,@#42    ;;ACT-11?
005624  001454                                  BEQ     69$             ;;BRANCH IF YES
005626  104401  005674                          TYPE    ,70$            ;;TYPE ASCIZ STRING
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
005632  005737  000042                          TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
005636  001012                                  BNE     71$             ;;BRANCH IF YES
005640  123727  001242  000001                  CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
005646  001406                                  BEQ     71$             ;;BRANCH IF YES
005650  023727  001154  000176                  CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
005656  001005                                  BNE     72$             ;;BRANCH IF NO
005660  104407                                  GTSWR                   ;;GET SOFT-SWR SETTINGS
005662  000403                                  BR      72$
005664  112737  000001  001150  71$:     MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
005672                                  72$:
005672  000431                                  BR      69$             ;;GET OVER THE ASCIZ
                                        ;;70$:  .ASCIZ  <CRLF>@DZRMDA - RM03 SUBSYSTEM FUNCTIONAL TEST, PART 2@<CRLF>
005756                                  69$:                                    ..


                                        ;FIND OUT IF PROGRAM IS RUNNING IN STANDALONE MODE
005756  005737  000042                          TST     42              ;IS LOC 42 ZERO ??
005762  001003                                  BNE     10$             ;NO - NOT IN STANDALONE
005764  105737  001242                          TSTB    $ENV            ;IS APT ENVIRONMENT ZERO ??
005770  001411                                  BEQ     STANDALONE      ;YES - PROGRAM IN STANDALONE
005772                                  10$:


                                        ;PROGRAM NOT RUNNING IN STANDALONE - SEE IF SIZING IS ALLOWED
005772  132737  000200  001243                  BITB    #BIT7,$ENVM     ;SIZING ALLOWED ??
006000  001003                                  BNE     20$             ;NO
006002  012737  000377  001300                  MOV     #377,$DEVM      ;YES - SET DEVICE MAP FOR ALL DEVICES
006010                                  20$:


                                        ;GO TO COMMON START CODE
006010  000137  006632                          JMP     CMNSTART
```

```
                                STANDALONE:
006014  004737  060652                JSR     PC,$TKINT       ;INITIALIZE CONSOLE

                                ;SEE IF THIS IS THE FIRST START AFTER PROGRAM WAS LOADED
006020  005327  000001                DEC     #1              ;FIRST START ??
006024  100024                        BPL     10$             ;YES !!


                                ;SEE IF THE USER WANTS TO KEEP SAME DEVICES FOR TESTING
006026  104401  063204        5$:     TYPE    ,CNSL00         ;MAINTAIN PREVIOUS PARAMETERS??
006032  104411                        RDCHR                   ;GET RESPONSE
006034  012637  001176                MOV     (SP)+,$TMP1     ;ECHO RESPONSE
006040  104401  001176                TYPE    ,$TMP1
006044  123727  001176  000131        CMPB    $TMP1,#'Y       ;YES RESPONSE??
006052  001002                        BNE     6$              ;NO!!
006054  000137  006764                JMP     READY           ;KEEP PREVIOUS PARAMETERS
006060  123727  001176  000116  6$:   CMPB    $TMP1,#'N       ;NO RESPONSE??
006066  001420                        BEQ     20$             ;GET NEW PARAMETERS
006070  104401  063053                TYPE    ,QSTMRK         ;NOT YES OR NO, TYPE "?"
006074  000754                        BR      5$              ;RETRY
006076                          10$:

                                ;SEE IF OPERATOR WANTS HELP FILE
006076  104401  063055                TYPE    ,HELPQST        ;WANT HELP ??
006102  104411                        RDCHR                   ;GET RESPONSE
006104  012637  001176                MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
006110  104401  001176                TYPE    ,$TMP1
006114  123727  001176  000131        CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
006122  001002                        BNE     20$             ;NO - DONT TYPE HELP
006124  104401  102574                TYPE    ,HELP           ;YES - TYPE HELP TEXT
006130                          20$:

                                ;SEE IF USER WANTS TO CHANGE RM03 UNIBUS ADDRESS
006130  104401  063111                TYPE    ,UBUSQST        ;WANT TO CHANGE ADDRESS ??
006134  104411                        RDCHR                   ;GET RESPONSE
006136  012637  001176                MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
006142  104401  001176                TYPE    ,$TMP1
006146  123727  001176  000131        CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ??
006154  001137                        BNE     90$             ;NO !!
006156                          30$:

                                ;DIALOGUE TO CHANGE THE UNIBUS ADDRESS, AND INTERRUPT VECTOR
006156  104401  063243                TYPE    ,CNSL01         ;TYPE CURRENT BUS ADDRESS
006162  013746  001276                MOV     $BASE,-(SP)     ;;SAVE $BASE FOR TYPEOUT
006166  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
006170  104401  063044                TYPE    ,CLSPRN
006174  104413                        RDOCT                   ;GET NEW BUS ADDRESS
006176  012637  001176                MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
006202  001412                        BEQ     50$             ;YES-SKIP TO NEXT ENTRY
006204  022737  160000  001176        CMP     #160000,$TMP1   ;BASE ADDRESS IN I/O PAGE??
006212  101403                        BLOS    40$             ;YES
006214  104401  063270                TYPE    ,CNSL02         ;TYPE WARNING MESSAGE
006220  000756                        BR      30$             ;RETRY
006222  013737  001176  001276  40$:   MOV     $TMP1,$BASE     ;STORE NEW BUS ADDRESS
006230  113737  001272  001176  50$:   MOVB    $VECT1,$TMP1    ;TYPE CURRENT VECTOR ADDRESS
006236  105037  001177                CLRB    $TMP1+1
```

```
3372  006242  104401  063351                      TYPE    ,CNSL03
3373  006246  013746  001176                      MOV     $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
3374  006252  104403                              TYPOS                   ;;GO TYPE--OCTAL ASCII
3375  006254     003                              .BYTE   3               ;;TYPE 3 DIGIT(S)
3376  006255     000                              .BYTE   0               ;;SUPPRESS LEADING ZEROS
3377  006256  104401  063044                      TYPE    ,CLSPRN
3378  006262  104413                              RDOCT                   ;GET NEW VECTOR ADDRESS
3379  006264  012637  001176                      MOV     (SP)+,$TMP1     ;CARRIAGE RETURN?
3380  006270  001412                              BEQ     70$             ;YES-SKIP TO NEXT ENTRY
3381  006272  022737  001000  001176              CMP     #1000,$TMP1     ;VECTOR ADDRESS < 1000??
3382  006300  101003                              BHI     60$             ;YES!!
3383  006302  104401  063401                      TYPE    ,CNSL04         ;TYPE WARNING MESSAGE
3384  006306  000750                              BR      50$             ;RETRY
3385  006310  113737  001176  001272  60$:       MOVB    $TMP1,$VECT1    ;STORE NEW VECTOR ADDRESS
3386  006316  113737  001273  001176  70$:       MOVB    $VECT1+1,$TMP1  ;TYPE CURRENT PRIORITY
3387  006324  006237  001176                      ASR     $TMP1
3388  006330  006237  001176                      ASR     $TMP1
3389  006334  006237  001176                      ASR     $TMP1
3390  006340  006237  001176                      ASR     $TMP1
3391  006344  006237  001176                      ASR     $TMP1
3392  006350  105037  001177                      CLRB    $TMP1+1
3393  006354  104401  063455                      TYPE    ,CNSL05
3394  006360  013746  001176                      MOV     $TMP1,-(SP)     ;;SAVE $TMP1 FOR TYPEOUT
3395  006364  104403                              TYPOS                   ;;GO TYPE--OCTAL ASCII
3396  006366     001                              .BYTE   1               ;;TYPE 1 DIGIT(S)
3397  006367     000                              .BYTE   0               ;;SUPPRESS LEADING ZEROS
3398  006370  104401  063044                      TYPE    ,CLSPRN
3399  006374  104413                              RDOCT                   ;GET NEW PRIORITY
3400  006376  012637  001176                      MOV     (SP)+,$TMP1     ;CARRIAGE RETURN??
3401  006402  001424                              BEQ     90$             ;YES-SKIP TO NEXT ENTRY
3402  006404  023727  001176  000007              CMP     $TMP1,#7        ;LEGAL PRIORITY??
3403  006412  002403                              BLT     80$             ;YES!!
3404  006414  104401  063511                      TYPE    ,CNSL06         ;TYPE WARNING MESSAGE
3405  006420  000736                              BR      70$             ;RETRY
3406  006422                          80$:                                ;STORE NEW PRIORITY
3407  006422  006337  001176                      ASL     $TMP1
3408  006426  006337  001176                      ASL     $TMP1
3409  006432  006337  001176                      ASL     $TMP1
3410  006436  006337  001176                      ASL     $TMP1
3411  006442  006337  001176                      ASL     $TMP1
3412  006446  113737  001176  001273              MOVB    $TMP1,$VECT1+1
3413  006454                          90$:
3414
3415                                  ;DIALOGUE TO INPUT DEVICE NUMBERS
3416  006454  005037  001300                      CLR     $DEVM           ;CLEAR DEVICE MAP
3417  006460  104401  063536                      TYPE    ,CNSL07         ;TYPE INPUT INSTRUCTIONS
3418  006464  104401  063047                      TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
3419  006470  104411                              RDCHR                   ;GET RESPONSE
3420  006472  012637  001176                      MOV     (SP)+,$TMP1     ;ECHO RESPONSE
3421  006476  10440.  001176                      TYPE    ,$TMP1
3422  006502  023727  001176  000101              CMP     $TMP1,#'A       ;TEST ALL DRIVES??
3423  006510  001017                              BNE     110$            ;NO
3424  006512  012737  000377  001300              MOV     #377,$DEVM      ;TEST ALL DEVICES
3425  006520  000444                              BR      140$            ;SKIP TO NEXT ENTRY
3426  006522  104401  063047          100$:       TYPE    ,PROMPT         ;TYPE PROMPTING CHARACTER
3427  006526  104411                              RDCHR                   ;GET RESPONSE
```

```
3428  006530  012637  001176                    MOV     (SP)+,$TMP1     ;ECHO RESPONSE
3429  006534  104401  001176                    TYPE    ,$TMP1
3430  006540  023727  001176  000015            CMP     $TMP1,#CR       ;CARRIAGE RETURN??
3431  006546  001431                            BEQ     140$
3432  006550  023727  001176  000060  110$:     CMP     $TMP1,#'0       ;NUMBER < 0??
3433  006556  002404                            BLT     120$            ;YES!!
3434  006560  023727  001176  000067            CMP     $TMP1,#'7       ;NUMBER > 7??
3435  006566  003403                            BLE     130$            ;NO!!
3436  006570  104401  063053          120$:     TYPE    ,QSTMRK         ;TYPE "?"
3437  006574  000752                            BR      100$            ;RETRY
3438  006576  013701  001176          130$:     MOV     $TMP1,R1        ;R1=DRIVE NUMBER
3439  006602  042701  177770                    BIC     #^C7,R1
3440  006606  116102  063776                    MOVB    ATNTBL(R1),R2           ;DECODE DEVICE NUMBER
3441  006612  042702  177400                    BIC     #^C377,R2               ;CLEAR UNUSED BITS
3442  006616  050237  001300                    BIS     R2,$DEVM                ;SET DEVICE # IN MAP
3443  006622  122737  000377  001300            CMPB    #377,$DEVM      ;DONE ??
3444  006630  101334                            BHI     100$            ;NO
3445  006632                          140$:
3446
```

GET VALUE FOR SOFTWARE SWITCH REGISTER

```
3447  006632                                  CMNSTART:
3448
3449                                          ;ASSEMBLE TEST QUE FROM DEVICE MAP
3450  006632  013700  001300                          MOV     $DEVM,R0        ;R0 = DEVICE MAP
3451  006636  012701  001450                          MOV     #TSTQUE+2,R1    ;R1 = ADDRESS OF FIRST ENTRY IN QUE
3452  006642  010137  001446                          MOV     R1,TSTQUE       ;INITIALIZE ENTRY POINTER
3453  006646  012702  000001                          MOV     #1,R2           ;R2 = DEVICE POINTER
3454  006652  005003                                  CLR     R3              ;R3 = DEVICE NUMBER
3455  006654  030200                          10$:    BIT     R2,R0           ;IS THIS DEVICE IN MAP ??
3456  006656  001406                                  BEQ     20$             ;NO !!
3457  006660  010311                                  MOV     R3,(R1)         ;YES - ENTER DEVICE NUMBER IN QUE
3458  006662  116361  063776  000001                  MOVB    ATNTBL(R3),1(R1);ENTER ATTENTION BIT IN QUE
3459  006670  062701  000002                          ADD     #2,R1           ;ADVANCE ENTRY POINTER
3460  006674  006302                          20$:    ASL     R2              ;ADVANCE DEVICE POINTER
3461  006676  105702                                  TSTB    R2              ;DONE ALL DEVICES ??
3462  006700  001402                                  BEQ     25$             ;YES
3463  006702  005203                                  INC     R3              ;ADVANCE DEVICE NUMBER
3464  006704  000763                                  BR      10$             ;ENTER NEXT DEVICE
3465  006706  005011                          25$:    CLR     (R1)            ;TERMINATE TEST QUE
3466
3467                                          ;SIZE FOR CLOCK
3468  006710  004737  037500                          JSR     PC,SIZCLK       ;SEE IF CLOCK PRESENT
3469  006714  000403                                  BR      40$             ;YES - CLOCK IS PRESENT
3470  006716  104000                          30$:    ERROR                   ;NO CLOCK
3471  006720  000000                                  HALT
3472  006722  000775                                  BR      30$
3473  006724                                  40$:
3474                                          .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
3475  006724  005737  000042                          TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
3476  006730  001012                                  BNE     64$             ;;BRANCH IF YES
3477  006732  123727  001242  000001                  CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
3478  006740  001406                                  BEQ     64$             ;;BRANCH IF YES
3479  006742  023727  001154  000176                  CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
3480  006750  001005                                  BNE     65$             ;;BRANCH IF NO
3481  006752  104407                                  GTSWR                   ;;GET SOFT-SWR SETTINGS
3482  006754  000403                                  BR      65$
3483  006756  112737  000001  001150  64$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
3484  006764                                  65$:
3485  006764  000240                          READY:  NOP                     ;READY TO START TEST
3486  006766  004737  060652                          JSR     PC,$TKINT       ;INITIALIZE TTY
3487  006772  013746  000300                          MOV     PR6,-(SP)       ;;PUT NEW PS ON STACK
3488  006776  012746  007004                          MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
3489  007002  000002                                  RTI                     ;;POP NEW PC AND PS
3490  007004                                  64$:
3491  007004  117737  172436  001234                  MOVB    @TSTQUE,$UNIT   ;LOAD UNIT NUMBER
3492  007012  005037  001472                          CLR     MEDENB          ;CLEAR MEDIA ENABLE
```

```
                                ;;*****************************************************************
                                ;*TEST 1         CONTROLLER ACCESS TEST
                                ;;*****************************************************************
007016                          TST1:
007016  000240                          NOP                             ;START OF TEST
007020  012737  007034  001122          MOV     #1$,$LPADR
007026  012737  007034  001124          MOV     #1$,$LPERR
007034                          1$:
007034  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
007040  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
007044  013701  001446                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
007050  012737  000001  001226          MOV     #1,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
007056  005001                          CLR     R1
007060  013746  000004                  MOV     ERRVEC,-(SP)            ;;PUSH ERRVEC ON STACK
007064  013746  000006                  MOV     ERRVEC+2,-(SP)                  ;;PUSH ERRVEC+2 ON STACK
007070  012737  007162  000004          MOV     #3$,ERRVEC
007076  012737  000300  000006          MOV     #PR6,ERRVEC+2

007104  110160  000001                  MOVB    R1,RMCS1+1(R0)          ;MOVE HI BYTE TO RMCS1
007110  010160  000002                  MOV     R1,RMWC(R0)             ;MOVE WORD COUNT REGISTER
007114  016002  000002                  MOV     RMWC(R0),R2
007120  010160  000004                  MOV     R1,RMBA(R0)             ;MOVE BUS ADDRESS REGISTER
007124  016002  000004                  MOV     RMBA(R0),R2
007130  010160  000010                  MOV     R1,RMCS2(R0)            ;MOVE CONTROL STATUS REGISTER
007134  016002  000010                  MOV     RMCS2(R0),R2
007140  010160  000022                  MOV     R1,RMDB(R0)             ;MOVE DATA BUFFER
007144  016002  000022                  MOV     RMDB(R0),R2
007150  012637  000006                  MOV     (SP)+,ERRVEC+2                  ;;POP STACK INTO ERRVEC+2
007154  012637  000004                  MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
007160  000415                          BR      7$                      ;NO BUS TIMEOUT OCCURRED

007162  022626                  3$:     CMP     (SP)+,(SP)+             ;ADJUST STACK
007164  012637  000006                  MOV     (SP)+,ERRVEC+2                  ;;POP STACK INTO ERRVEC+2
007170  012637  000004                  MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
007174  104110                          ERROR   110                     ;CANNOT ACCESS MASSBUS CONTROLLER
007176  005737  000042                  TST     42                      ;STAND ALONE MODE??
007202  001002                          BNE     5$                      ;NO!!
007204  000137  005322                  JMP     START                   ;YES-GO GET $BASE
007210  000137  032262          5$:     JMP     $EOP                    ;GO TO END OF PASS HANDLER
007214                          7$:

                                ;;*****************************************************************
                                ;*TEST 2         DEVICE AVAILABLE TEST

                                ;;*****************************************************************
007214                          TST2:
007214  000004                          SCOPE                           ;SCOPE CALL
007216  000240                          NOP                             ;START OF TEST
007220  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
007224  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
007230  013701  001446                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
007234  012737  000002  001226          MOV     #2,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX

007242  004737  046154                  JSR     PC,CNTCLR
007246  000404                          BR      2$      ;GO TO 2$ IF NO ERROR
007250  000240                          NOP                             ;RETURN HERE IF ERROR
```

```
                                        ERROR                           ;ERROR NUMBER DEFINED BY SUB
                                        JMP     7$      ;GO TO 7$ IF ERROR WAS FOUND
                                2$:
                                        MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
                                        MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
                                        MOV     #5$,ERRVEC
                                        MOV     PR6,ERRVEC+2

                                        MOV     RMCS1(R0),$TMP1 ;GET DVA STATUS
                                        MOV     RMCS2(R0),$TMP0 ;GET NED STATUS
                                        MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
                                        MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
                                        BIT     #NED,$TMP0      ;NONEXISTENT DEVICE??
                                        BEQ     3$              ;NO!!
                                        ERROR   111             ;NONEXISTENT DEVICE
                                        BR      7$
                                3$:     BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
                                        BNE     9$              ;YES!!
                                        ERROR   112             ;DEVICE NOT AVAILABLE
                                        BR      7$

                                5$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
                                        MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
                                        MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
                                        ERROR   113             ;BUS TIMEOUT (04 TRAP)
                                7$:     JMP     $EOSP

                                9$:

                                ;;*****************************************************************
                                ;*TEST 3        DRIVE TYPE TEST
                                ;;*****************************************************************
                                TST3:
                                        SCOPE                           ;SCOPE CALL
                                        NOP                             ;START OF TEST
                                        MOV     #STACK,SP               ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
                                        MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
                                        MOV     #3,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX

                                        JSR     PC,CNTCLR
                                        BR      2$      ;GO TO 2$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR NUMBER DEFINED BY SUB
                                        JMP     4$      ;GO TO 4$ IF ERROR WAS FOUND
                                2$:
                                        MOVB    #RMDT,GETINX    ;SETUP GET INDEX TABLE
                                        MOVB    #200,GETINX+1
                                        MOV     #5$,RMDTI               ;RMDT INPUT BUFFER = 5$
                                        JSR     PC,GET                  ;GO GET DRIVE TYPE
                                        BR      3$      ;GO TO 3$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR NUMBER SPECIFIED BY GET
                                3$:     CMP     #SNGPRT,RMDTI   ;SINGLE PORT RM03??
                                        BEQ     5$              ;YES!!
```

```
                        T3      DRIVE TYPE TEST

3605  007510  022737  024024  001354        CMP     #DULPRT,RMDTI   ;DUAL PORT RM03??
3606  007516  001411                        BEQ     5$              ;YES!!
3607  007520  012737  020024  001176        MOV     #SNGPRT,$TMP1
3608  007526  012737  024024  001200        MOV     #DULPRT,$TMP2
3609  007534  104114                        ERROR   114             ;NOT AN RM03
3610  007536  000137  032226        4$:     JMP     $EOSP           ;GO TO SUBPASS HANDLER.
3611
3612  007542                        5$:
3613                                ;;******************************************************************
3614                                ;*TEST 4        FORMAT ZEROS - 18
3615                                ;;******************************************************************
3616  007542                        TST4:
3617  007542  000004                        SCOPE                   ;SCOPE CALL
3618  007544  000240                        NOP                     ;START OF TEST
3619  007546  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
3620  007552  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
3621  007556  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
3622  007562  012737  000004  001226        MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3623  007570                        10$:
3624
3625                                ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
3626  007570  012737  000000  001432        MOV     #0.RMDC0                ;CYLINDER = 0
3627  007576  012737  000000  001404        MOV     #0,RMDA0                ;TRACK = SECTOR = 0
3628  007604  012737  000000  001430        MOV     #0,RMOF0                ;18 BIT FORMAT
3629  007612  012737  177376  001400        MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
3630  007620  012737  102574  001402        MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
3631  007626  012737  000062  001376        MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
3632  007634  012737  064110  001174        MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
3633  007642  012737  000001  001176        MOV     #1,$TMP1
3634  007650  004737  036114                JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
3635  007654                        20$:
3636
3637                                ;PREPARE DEVICE FOR DATA TRANSFER
3638  007654  004737  033274                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
3639  007660  154130                        .WORD   154130  ;TASK DESCRIPTOR
3640  007662  000404                        BR      30$                     ;GO TO 30$ IF NO ERROR
3641  007664  000240                        NOP                     ;RETURN HERE IF ERROR
3642  007666  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3643  007670  000137  010342                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3644  007674                        30$:
3645
3646                                ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
3647  007674  012737  000005  001376        MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
3648  007702  012702  001533                MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
3649  007706  112722  000006                MOVB    #RMDA,(R2)+
3650  007712  112722  000034                MOVB    #RMDC,(R2)+
3651  007716  112722  000032                MOVB    #RMOF,(R2)+
3652  007722  112722  000000                MOVB    #RMCS1,(R2)+
3653  007726  112722  000200                MOVB    #200,(R2)+
3654  007732  004737  037262                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
3655  007736  000404                        BR      40$     ;GO TO 40$ IF NO ERROR
3656  007740  000240                        NOP                     ;RETURN HERE IF ERROR
3657  007742  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
3658  007744  000137  010342                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3659  007750                        40$:
3660
```

```
3661                                    ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
3662  007750  004737  036726                      JSR     PC,GETSTS               ;SETUP FOR STATUS
3663  007754  004737  037622                      JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
3664  007760                                50$:
3665
3666                                    ;GO READ SEEK STATUS
3667  007760  004737  037012                      JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
3668  007764  000404                              BR      60$     ;GO TO 60$ IF NO ERROR
3669  007766  000240                              NOP                             ;RETURN HERE IF ERROR
3670  007770  104000                              ERROR                           ;ERROR DEFINED BY GET SUB
3671  007772  000137  010342                      JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3672  007776                                60$:
3673
3674                                    ;VERIFY THE RESULTS OF THE SEEK COMMAND
3675  007776  004737  044714                      JSR     PC,SEKSTS               ;GO VERIFY RESULTS OF SEEK OPERATION
3676  010002  000405                              BR      70$                             ;GO TO 70$ IF NO ERROR
3677  010004  000240                              NOP                             ;RETURN HERE IF ERROR
3678  010006  104000                              ERROR                           ;ERROR # DEFINED BY SEKSTS SUBROUTINE
3679  010010  004736                              JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
3680  010012  000137  010342                      JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
3681  010016                                70$:
3682
3683                                    ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
3684  010016  012737  000063  001376              MOV     #WH!GO,RMCS1            ;LOAD WRITE HEADER AND DATA
3685  010024  012702  001536                      MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
3686  010030  112722  000002                      MOVB    #RMWC,(R2)+
3687  010034  112722  000004                      MOVB    #RMBA,(R2)+
3688  010040  112722  000000                      MOVB    #RMCS1,(R2)+
3689  010044  112722  000200                      MOVB    #200,(R2)+
3690  010050  004737  037262                      JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
3691  010054  000404                              BR      80$     ;GO TO 80$ IF NO ERROR
3692  010056  000240                              NOP                             ;RETURN HERE IF ERROR
3693  010060  104000                              ERROR                           ;ERROR DEFINED BY PUT SUB
3694  010062  000137  010342                      JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3695  010066                                80$:
3696
3697                                    ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
3698  010066  004737  037622                      JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
3699  010072  004737  037012                      JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
3700  010076  000404                              BR      90$     ;GO TO 90$ IF NO ERROR
3701  010100  000240                              NOP                             ;RETURN HERE IF ERROR
3702  010102  104000                              ERROR                           ;ERROR DEFINED BY GET SUB
3703  010104  000137  010342                      JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3704  010110                                90$:
3705
3706                                    ;VERIFY RESULTS OF WRITE COMMAND
3707  010110  004737  040006                      JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
3708  010114  000405                              BR      100$                            ;GO TO 100$ IF NO ERROR
3709  010116  000240                              NOP                             ;RETURN HERE IF ERROR
3710  010120  104000                              ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
3711  010122  004736                              JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
3712  010124  000137  010342                      JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
3713  010130                                100$:
3714  010130  004737  052312                      JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
3715  010134  000405                              BR      110$                            ;GO TO 110$ IF NO ERROR
3716  010136  000240                              NOP                             ;RETURN HERE IF ERROR
```

```
3717  010140  104000                        ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
3718  010142  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
3719  010144  000137  010342                JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
3720  010150                          110$:
3721  010150  004737  040640                JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
3722  010154  000405                        BR      120$                            ;GO TO 120$ IF NO ERROR
3723  010156  000240                        NOP                             ;RETURN HERE IF ERROR
3724  010160  104000                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
3725  010162  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
3726  010164  000137  010342                JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
3727  010170                          120$:
3728
3729
3730                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
3731  010170  012737  000073  001376        MOV     #RH!GO,RMCS10                   ;READ HEADER & DATA COMMAND
3732  010176  012737  103600  001402        MOV     #BUFTWO,RMBAO                   ;CHANGE BUS ADDRESS
3733  010204  004737  037262                JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
3734  010210  000404                        BR      130$    ;GO TO 130$ IF NO ERROR
3735  010212  000240                        NOP                             ;RETURN HERE IF ERROR
3736  010214  104000                        ERROR                           ;ERROR DEFINED BY PUT SUB
3737  010216  000137  010342                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3738  010222                          130$:
3739
3740                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
3741  010222  004737  037622                JSR     PC,TIMOUT                       ;WAIT FOR READ TO COMPLETE
3742  010226  004737  037012                JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
3743  010232  000404                        BR      140$    ;GO TO 140$ IF NO ERROR
3744  010234  000240                        NOP                             ;RETURN HERE IF ERROR
3745  010236  104000                        ERROR                           ;ERROR DEFINED BY GET SUB
3746  010240  000137  010342                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3747  010244                          140$:
3748
3749                                  ;VERIFY THE RESULTS OF READ OPERATION
3750  010244  004737  040006                JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
3751  010250  000405                        BR      150$                            ;GO TO 150$ IF NO ERROR
3752  010252  000240                        NOP                             ;RETURN HERE IF ERROR
3753  010254  104000                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
3754  010256  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
3755  010260  000137  010342                JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
3756  010264                          150$:
3757  010264  004737  052312                JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
3758  010270  000405                        BR      160$                            ;GO TO 160$ IF NO ERROR
3759  010272  000240                        NOP                             ;RETURN HERE IF ERROR
3760  010274  104000                        ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
3761  010276  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
3762  010300  000137  010342                JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
3763  010304                          160$:
3764  010304  004737  040640                JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
3765  010310  000405                        BR      170$                            ;GO TO 170$ IF NO ERROR
3766  010312  000240                        NOP                             ;RETURN HERE IF ERROR
3767  010314  104000                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
3768  010316  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
3769  010320  000137  010342                JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
3770  010324                          170$:
3771
3772                                  ;VERIFY DATA
```

```
3773  010324  004737  036360          JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
3774  010330  102574                  .WORD   BUFONE          ;STARTING ADDRESS OF WRITE BUFFER
3775  010332  103600                  .WORD   BUFTWO          ;STARTING ADDRESS OF READ BUFFER
3776  010334  000402                  BR      180$                    ;GO TO 180$ IF NO ERROR
3777  010336  000240                  NOP                     ;RETURN HERE IF ERROR
3778  010340  104000                  ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
3779  010342                  180$:
3780
3781                            ;;*****************************************************************
3782                            ;*TEST 5         FORMAT ZEROS - 16
3783                            ;;*****************************************************************
3784  010342                  TST5:
3785  010342  000004                  SCOPE                   ;SCOPE CALL
3786  010344  000240                  NOP                     ;START OF TEST
3787  010346  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
3788  010352  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
3789  010356  013701  001446          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
3790  010362  012737  000005  001226  MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3791  010370                  10$:
3792
3793                            ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
3794  010370  012737  000000  001432  MOV     #0,RMDC0                ;CYLINDER = 0
3795  010376  012737  000000  001404  MOV     #0,RMDA0                ;TRACK = SECTOR = 0
3796  010404  012737  010000  001430  MOV     #FMT16,RMOF0            ;16 BIT FORMAT
3797  010412  012737  177376  001400  MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
3798  010420  012737  102574  001402  MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
3799  010426  012737  000062  001376  MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
3800  010434  012737  064110  001174  MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
3801  010442  012737  000001  001176  MOV     #1,$TMP1
3802  010450  004737  036114          JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
3803  010454                  20$:
3804
3805                            ;PREPARE DEVICE FOR DATA TRANSFER
3806  010454  004737  033274          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
3807  010460  154130                  .WORD   154130  ;TASK DESCRIPTOR
3808  010462  000404                  BR      30$                     ;GO TO 30$ IF NO ERROR
3809  010464  000240                  NOP                     ;RETURN HERE IF ERROR
3810  010466  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
3811  010470  000137  011142          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
3812  010474                  30$:
3813
3814                            ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
3815  010474  012737  000005  001376  MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
3816  010502  012702  001533          MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
3817  010506  112722  000006          MOVB    #RMDA,(R2)+
3818  010512  112722  000034          MOVB    #RMDC,(R2)+
3819  010516  112722  000032          MOVB    #RMOF,(R2)+
3820  010522  112722  000000          MOVB    #RMCS1,(R2)+
3821  010526  112722  000200          MOVB    #200,(R2)+
3822  010532  004737  037262          JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
3823  010536  000404                  BR      40$     ;GO TO 40$ IF NO ERROR
3824  010540  000240                  NOP                     ;RETURN HERE IF ERROR
3825  010542  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
3826  010544  000137  011142          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
3827  010550                  40$:
3828
```

```
                                ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
010550  004737  036726                  JSR     PC,GETSTS               ;SETUP FOR STATUS
010554  004737  037622                  JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
010560                          50$:

                                ;GO READ SEEK STATUS
010560  004737  037012                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
010564  000404                          BR      60$     ;GO TO 60$ IF NO ERROR
010566  000240                          NOP                     ;RETURN HERE IF ERROR
010570  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
010572  000137  011142                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
010576                          60$:

                                ;VERIFY THE RESULTS OF THE SEEK COMMAND
010576  004737  044714                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
010602  000405                          BR      70$                     ;GO TO 70$ IF NO ERROR
010604  000240                          NOP                     ;RETURN HERE IF ERROR
010606  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
010610  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
010612  000137  011142                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
010616                          70$:

                                ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
010616  012737  000063  001376          MOV     #WH!GO,RMCS1            ;LOAD WRITE HEADER AND DATA
010624  012702  001536                  MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
010630  112722  000002                  MOVB    #RMWC,(R2)+
010634  112722  000004                  MOVB    #RMBA,(R2)+
010640  112722  000000                  MOVB    #RMCS1,(R2)+
010644  112722  000200                  MOVB    #200,(R2)+
010650  004737  037262                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
010654  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
010656  000240                          NOP                     ;RETURN HERE IF ERROR
010660  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
010662  000137  011142                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
010666                          80$:

                                ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
010666  004737  037622                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
010672  004737  037012                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
010676  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
010700  000240                          NOP                     ;RETURN HERE IF ERROR
010702  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
010704  000137  011142                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
010710                          90$:

                                ;VERIFY RESULTS OF WRITE COMMAND
010710  004737  040006                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
010714  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
010716  000240                          NOP                     ;RETURN HERE IF ERROR
010720  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
010722  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
010724  000137  011142                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
010730                          100$:
010730  004737  052312                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
010734  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
010736  000240                          NOP                     ;RETURN HERE IF ERROR
```

```
3885  010740  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
3886  010742  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3887  010744  000137  011142                JMP     18$                     ;GO TO 18$ IF ERROR WAS FOUND
3888  010750                          11$:
3889  010750  004737  040640                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
3890  010754  000405                        BR      12$                     ;GO TO 12$ IF NO ERROR
3891  010756  000240                        NOP                     ;RETURN HERE IF ERROR
3892  010760  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
3893  010762  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3894  010764  000137  011142                JMP     18$                     ;GO TO 18$ IF ERROR WAS FOUND
3895  010770                          12$:
3896
3897
3898                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
3899  010770  012737  000073  001376        MOV     #RH!GO,RMCS1O           ;READ HEADER & DATA COMMAND
3900  010776  012737  103600  001402        MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
3901  011004  004737  037262                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
3902  011010  000404                        BR      13$     ;GO TO 13$ IF NO ERROR
3903  011012  000240                        NOP                     ;RETURN HERE IF ERROR
3904  011014  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
3905  011016  000137  011142                JMP     18$     ;GO TO 18$ IF ERROR WAS FOUND
3906  011022                          13$:
3907
3908                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
3909  011022  004737  037622                JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
3910  011026  004737  037012                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
3911  011032  000404                        BR      14$     ;GO TO 14$ IF NO ERROR
3912  011034  000240                        NOP                     ;RETURN HERE IF ERROR
3913  011036  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
3914  011040  000137  011142                JMP     18$     ;GO TO 18$ IF ERROR WAS FOUND
3915  011044                          14$:
3916
3917                                  ;VERIFY THE RESULTS OF READ OPERATION
3918  011044  004737  040006                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
3919  011050  000405                        BR      15$                     ;GO TO 15$ IF NO ERROR
3920  011052  000240                        NOP                     ;RETURN HERE IF ERROR
3921  011054  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
3922  011056  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3923  011060  000137  011142                JMP     18$                     ;GO TO 18$ IF ERROR WAS FOUND
3924  011064                          15$:
3925  011064  004737  052312                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
3926  011070  000405                        BR      16$                     ;GO TO 16$ IF NO ERROR
3927  011072  000240                        NOP                     ;RETURN HERE IF ERROR
3928  011074  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
3929  011076  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3930  011100  000137  011142                JMP     18$                     ;GO TO 18$ IF ERROR WAS FOUND
3931  011104                          16$:
3932  011104  004737  040640                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
3933  011110  000405                        BR      17$                     ;GO TO 17$ IF NO ERROR
3934  011112  000240                        NOP                     ;RETURN HERE IF ERROR
3935  011114  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
3936  011116  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
3937  011120  000137  011142                JMP     18$                     ;GO TO 18$ IF ERROR WAS FOUND
3938  011124                          17$:
3939
3940                                  ;VERIFY DATA
```

```
 3941  011124  004737  036360            JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
 3942  011130  102574                    .WORD   BUFONE          ;STARTING ADDRESS OF WRITE BUFFER
 3943  011132  103600                    .WORD   BUFTWO          ;STARTING ADDRESS OF READ BUFFER
 3944  011134  000402                    BR      180$                    ;GO TO 180$ IF NO ERROR
 3945  011136  000240                    NOP                     ;RETURN HERE IF ERROR
 3946  011140  104000                    ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
 3947  011142                    180$:
 3948
 3949                              ;;*****************************************************************
 3950                              ;*TEST 6         ZERO FILL TEST
 3951                              ;;*****************************************************************
 3952  011142                    ST6:
 3953  011142  000004                    SCOPE                   ;SCOPE CALL
 3954  011144  000240                    NOP                     ;START OF TEST
 3955  011146  012706  001100            MOV     #STACK,SP       ;INITIALIZE STACK POINTER
 3956  011152  013700  001276            MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 3957  011156  013701  001446            MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 3958  011162  012737  000006  001226    MOV     #6,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
 3959  011170                    10$:
 3960
 3961                              ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
 3962  011170  012737  000000  001432    MOV     #0,RMDC0                ;CYLINDER = 0
 3963  011176  012737  000000  001404    MOV     #0,RMDA0                ;TRACK = SECTOR = 0
 3964  011204  012737  010000  001430    MOV     #FMT16,RMOF0            ;16 BIT FORMAT
 3965  011212  012737  177376  001400    MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
 3966  011220  012737  102574  001402    MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
 3967  011226  012737  000062  001376    MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
 3968  011234  012737  064110  001174    MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
 3969  011242  012737  000001  001176    MOV     #1,$TMP1
 3970  011250  004737  036114            JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
 3971  011254                    20$:
 3972
 3973                              ;PREPARE DEVICE FOR DATA TRANSFER
 3974  011254  004737  033274            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 3975  011260  154130                    .WORD   154130  ;TASK DESCRIPTOR
 3976  011262  000404                    BR      30$                     ;GO TO 30$ IF NO ERROR
 3977  011264  000240                    NOP                     ;RETURN HERE IF ERROR
 3978  011266  104000                    ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 3979  011270  000137  011756            JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 3980  011274                    30$:
 3981
 3982                              ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
 3983  011274  012737  000005  001376    MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
 3984  011302  012702  001533            MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
 3985  011306  112722  000006            MOVB    #RMDA,(R2)+
 3986  011312  112722  000034            MOVB    #RMDC,(R2)+
 3987  011316  112722  000032            MOVB    #RMOF,(R2)+
 3988  011322  112722  000000            MOVB    #RMCS1,(R2)+
 3989  011326  112722  000200            MOVB    #200,(R2)+
 3990  011332  004737  037262            JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 3991  011336  000404                    BR      40$     ;GO TO 40$ IF NO ERROR
 3992  011340  000240                    NOP                     ;RETURN HERE IF ERROR
 3993  011342  104000                    ERROR                   ;ERROR DEFINED BY PUT SUB
 3994  011344  000137  011756            JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 3995  011350                    40$:
 3996
```

```
 3997
 3998  011350  004737  036726                ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
 3999  011354  004737  037622                        JSR     PC,GETSTS               ;SETUP FOR STATUS
 4000  011360                                        JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
 4001                                        50$:
 4002
 4003  011360  004737  037012                ;GO READ SEEK STATUS
 4004  011364  000404                              JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 4005  011366  000240                              BR      60$     ;GO TO 60$ IF NO ERROR
 4006  011370  104000                              NOP                     ;RETURN HERE IF ERROR
 4007  011372  000137  011756                      ERROR                   ;ERROR DEFINED BY GET SUB
 4008  011376                                      JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 4009                                        60$:
 4010
 4011  011376  004737  044714                ;VERIFY THE RESULTS OF THE SEEK COMMAND
 4012  011402  000405                              JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
 4013  011404  000240                              BR      70$                     ;GO TO 70$ IF NO ERROR
 4014  011406  104000                              NOP                     ;RETURN HERE IF ERROR
 4015  011410  004736                              ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
 4016  011412  000137  011756                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 4017  011416                                      JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 4018                                        70$:
 4019
 4020  011416  012737  177776  001400        ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
 4021  011424  012737  000063  001376              MOV     #<^C2+1>,RMWCO          ;FORMAT PARTIAL SECTOR
 4022  011432  012702  001536                      MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
 4023  011436  112722  000002                      MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
 4024  011442  112722  000004                      MOVB    #RMWC,(R2)+
 4025  011446  112722  000000                      MOVB    #RMBA,(R2)+
 4026  011452  112722  000200                      MOVB    #RMCS1,(R2)+
 4027  011456  004737  037262                      MOVB    #200,(R2)+
 4028  011462  000404                              JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 4029  011464  000240                              BR      80$     ;GO TO 80$ IF NO ERROR
 4030  011466  104000                              NOP                     ;RETURN HERE IF ERROR
 4031  011470  000137  011756                      ERROR                   ;ERROR DEFINED BY PUT SUB
 4032  011474                                      JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 4033                                        80$:
 4034
 4035  011474  004737  037622                ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
 4036  011500  004737  037012                      JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
 4037  011504  000404                              JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 4038  011506  000240                              BR      90$     ;GO TO 90$ IF NO ERROR
 4039  011510  104000                              NOP                     ;RETURN HERE IF ERROR
 4040  011512  000137  011756                      ERROR                   ;ERROR DEFINED BY GET SUB
 4041  011516                                      JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 4042                                        90$:
 4043
 4044  011516  004737  040006                ;VERIFY RESULTS OF WRITE COMMAND
 4045  011522  000405                              JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 4046  011524  000240                              BR      100$                    ;GO TO 100$ IF NO ERROR
 4047  011526  104000                              NOP                     ;RETURN HERE IF ERROR
 4048  011530  004736                              ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 4049  011532  000137  011756                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 4050  011536                                      JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 4051  011536  004737  052312                100$:
 4052  011542  000405                              JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
                                                   BR      110$                    ;GO TO 110$ IF NO ERROR
```

```
 4053  011544  000240                        NOP                     ;RETURN HERE IF ERROR
 4054  011546  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 4055  011550  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 4056  011552  000137  011756                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 4057  011556                          110$:
 4058  011556  004737  040640                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 4059  011562  000405                        BR      120$                    ;GO TO 120$ IF NO ERROR
 4060  011564  000240                        NOP                     ;RETURN HERE IF ERROR
 4061  011566  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 4062  011570  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 4063  011572  000137  011756                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 4064  011576                          120$:
 4065
 4066
 4067                                    ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
 4068  011576  012737  177376  001400        MOV     #<^C<2+256.>+1>,RMWCO   ;READ A FULL SECTOR
 4069  011604  012737  000073  001376        MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
 4070  011612  012737  103600  001402        MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
 4071  011620  004737  037262                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 4072  011624  000404                        BR      130$    ;GO TO 130$ IF NO ERROR
 4073  011626  000240                        NOP                     ;RETURN HERE IF ERROR
 4074  011630  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
 4075  011632  000137  011756                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 4076  011636                          130$:
 4077
 4078                                    ;WAIT FOR READ TO COMPLETE AND GET STATUS
 4079  011636  004737  037622                JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
 4080  011642  004737  037012                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 4091  011646  000404                        BR      140$    ;GO TO 140$ IF NO ERROR
 4082  011650  000240                        NOP                     ;RETURN HERE IF ERROR
 4083  011652  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
 4084  011654  000137  011756                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 4085  011660                          140$:
 4086
 4087                                    ;VERIFY THE RESULTS OF READ OPERATION
 4088  011660  004737  040006                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 4089  011664  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
 4090  011666  000240                        NOP                     ;RETURN HERE IF ERROR
 4091  011670  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 4092  011672  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 4093  011674  000137  011756                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 4094  011700                          150$:
 4095  011700  004737  052312                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 4096  011704  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
 4097  011706  000240                        NOP                     ;RETURN HERE IF ERROR
 4098  011710  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 4099  011712  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 4100  011714  000137  011756                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 4101  011720                          160$:
 4102  011720  004737  040640                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 4103  011724  000405                        BR      170$                    ;GO TO 170$ IF NO ERROR
 4104  011726  000240                        NOP                     ;RETURN HERE IF ERROR
 4105  011730  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 4106  011732  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 4107  011734  000137  011756                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 4108  011740                          170$:
```

```
                                ;VERIFY DATA
011740  004737  036360                JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
011744  102574                        .WORD   BUFONE          ;STARTING ADDRESS OF WRITE BUFFER
011746  103600                        .WORD   BUFTWO          ;STARTING ADDRESS OF READ BUFFER
011750  000402                        BR      180$                    ;GO TO 180$ IF NO ERROR
011752  000240                        NOP                     ;RETURN HERE IF ERROR
011754  104000                        ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
011756                          180$:

                                ;;*****************************************************************
                                ;*TEST 7         FORMAT CHECK ZEROS - 16
                                ;;*****************************************************************
011756                          TST7:
011756  000004                        SCOPE                   ;SCOPE CALL
011760  000240                        NOP                     ;START OF TEST
011762  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
011766  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
011772  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
011776  012737  000007  001226        MOV     #7,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
012004                          10$:

                                ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
012004  012737  000000  001432        MOV     #0,RMDC0                ;CYLINDER = 0
012012  012737  000000  001404        MOV     #0,RMDA0                ;TRACK = SECTOR = 0
012020  012737  010000  001430        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
012026  012737  177376  001400        MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
012034  012737  102574  001402        MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
012042  012737  000062  001376        MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
012050  012737  064110  001174        MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
012056  012737  000001  001176        MOV     #1,$TMP1
012064  004737  036114                JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
012070                          20$:

                                ;PREPARE DEVICE FOR DATA TRANSFER
012070  004737  033274                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
012074  154130                        .WORD   154130  ;TASK DESCRIPTOR
012076  000404                        BR      30$                     ;GO TO 30$ IF NO ERROR
012100  000240                        NOP                     ;RETURN HERE IF ERROR
012102  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
012104  000137  012532                JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
012110                          30$:

                                ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
012110  012737  000005  001376        MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
012116  012702  001533                MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
012122  112722  000006                MOVB    #RMDA,(R2)+
012126  112722  000034                MOVB    #RMDC,(R2)+
012132  112722  000032                MOVB    #RMOF,(R2)+
012136  112722  000000                MOVB    #RMCS1,(R2)+
012142  112722  000200                MOVB    #200,(R2)+
012146  004737  037262                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
012152  000404                        BR      40$     ;GO TO 40$ IF NO ERROR
012154  000240                        NOP                     ;RETURN HERE IF ERROR
012156  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
012160  000137  012532                JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
```

```
012164                                  40$:
                                        ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
012164  004737  036726                          JSR     PC,GETSTS               ;SETUP FOR STATUS
012170  004737  037622                          JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
012174                                  50$:
                                        ;GO READ SEEK STATUS
012174  004737  037012                          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
012200  000404                                  BR      60$     ;GO TO 60$ IF NO ERROR
012202  000240                                  NOP                     ;RETURN HERE IF ERROR
012204  104000                                  ERROR                   ;ERROR DEFINED BY GET SUB
012206  000137  012532                          JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
012212                                  60$:
                                        ;VERIFY THE RESULTS OF THE SEEK COMMAND
012212  004737  044714                          JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
012216  000405                                  BR      70$                     ;GO TO 70$ IF NO ERROR
012220  000240                                  NOP                     ;RETURN HERE IF ERROR
012222  104000                                  ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
012224  004736                                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
012226  000137  012532                          JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
012232                                  70$:
                                        ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
012232  012737  000063  001376                  MOV     #WH!GO,RMCS1O           ;LOAD WRITE HEADER AND DATA
012240  012702  001536                          MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
012244  112722  000002                          MOVB    #RMWC,(R2)+
012250  112722  000004                          MOVB    #RMBA,(R2)+
012254  112722  000000                          MOVB    #RMCS1,(R2)+
012260  112722  000200                          MOVB    #200,(R2)+
012264  004737  037262                          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
012270  000404                                  BR      80$     ;GO TO 80$ IF NO ERROR
012272  000240                                  NOP                     ;RETURN HERE IF ERROR
012274  104000                                  ERROR                   ;ERROR DEFINED BY PUT SUB
012276  000137  012532                          JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
012302                                  80$:
                                        ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
012302  004737  037622                          JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
012306  004737  037012                          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
012312  000404                                  BR      90$     ;GO TO 90$ IF NO ERROR
012314  000240                                  NOP                     ;RETURN HERE IF ERROR
012316  104000                                  ERROR                   ;ERROR DEFINED BY GET SUB
012320  000137  012532                          JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
012324                                  90$:
                                        ;VERIFY RESULTS OF WRITE COMMAND
012324  004737  040006                          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
012330  000405                                  BR      100$                    ;GO TO 100$ IF NO ERROR
012332  000240                                  NOP                     ;RETURN HERE IF ERROR
012334  104000                                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
012336  004736                                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
012340  000137  012532                          JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
012344                                  100$:
012344  004737  052312                          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
```

```
4221  012350  000405                        BR      110$                    ;GO TO 110$ IF NO ERROR
4222  012352  000240                        NOP                     ;RETURN HERE IF ERROR
4223  012354  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4224  012356  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4225  012360  000137  012532                JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4226  012364                          110$:
4227  012364  004737  040640                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4228  012370  000405                        BR      120$                    ;GO TO 120$ IF NO ERROR
4229  012372  000240                        NOP                     ;RETURN HERE IF ERROR
4230  012374  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4231  012376  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4232  012400  000137  012532                JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4233  012404                          120$:
4234
4235
4236                                    ;WRITE CHECK HEADER AND DATA FOR SECTOR JUST WRITTEN
4237  012404  012737  000053  001376        MOV     #WCH!GO,RMCS1           ;WRITE CHECK COMMAND
4238  012412  004737  037262                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4239  012416  000404                        BR      130$    ;GO TO 130$ IF NO ERROR
4240  012420  000240                        NOP                     ;RETURN HERE IF ERROR
4241  012422  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
4242  012424  000137  012532                JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4243  012430                          130$:
4244
4245                                    ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
4246  012430  004737  037622                JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
4247  012434  004737  037012                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4248  012440  000404                        BR      140$    ;GO TO 140$ IF NO ERROR
4249  012442  000240                        NOP                     ;RETURN HERE IF ERROR
4250  012444  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
4251  012446  000137  012532                JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4252  012452                          140$:
4253
4254                                    ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
4255  012452  004737  040006                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4256  012456  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
4257  012460  000240                        NOP                     ;RETURN HERE IF ERROR
4258  012462  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4259  012464  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4260  012466  000137  012532                JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4261  012472                          150$:
4262  012472  004737  052312                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4263  012476  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
4264  012500  000240                        NOP                     ;RETURN HERE IF ERROR
4265  012502  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4266  012504  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4267  012506  000137  012532                JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4268  012512                          160$:
4269  012512  004737  040640                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4270  012516  000405                        BR      170$                    ;GO TO 170$ IF NO ERROR
4271  012520  000240                        NOP                     ;RETURN HERE IF ERROR
4272  012522  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4273  012524  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4274  012526  000137  012532                JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4275  012532                          170$:
4276
```

```
 4277  012532                                  260$:
 4278
 4279                                          ;;*****************************************************************
 4280                                          ;*TEST 10       FORMAT CHECK ZEROS W/ WCE ERROR
 4281                                          ;;*****************************************************************
 4282  012532                                  TST10:
 4283  012532  000004                                 SCOPE                   ;SCOPE CALL
 4284  012534  000240                                 NOP                     ;START OF TEST
 4285  012536  012706  001100                         MOV     #STACK,SP       ;INITIALIZE STACK POINTER
 4286  012542  013700  001276                         MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 4287  012546  013701  001446                         MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 4288  012552  012737  000010  001226                 MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 4289  012560                                  10$:
 4290
 4291                                          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
 4292  012560  012737  000000  001432                 MOV     #0,RMDC0                ;CYLINDER = 0
 4293  012566  012737  000000  001404                 MOV     #0,RMDA0                ;TRACK = SECTOR = 0
 4294  012574  012737  010000  001430                 MOV     #FMT16,RMOF0            ;16 BIT FORMAT
 4295  012602  012737  177376  001400                 MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
 4296  012610  012737  102574  001402                 MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
 4297  012616  012737  000062  001376                 MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
 4298  012624  012737  064110  001174                 MOV     #ZEROS,$TMP0            ;USE ALL ONES DATA PATTERN
 4299  012632  012737  000001  001176                 MOV     #1,$TMP1
 4300  012640  004737  036114                         JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
 4301  012644                                  20$:
 4302
 4303                                          ;PREPARE DEVICE FOR DATA TRANSFER
 4304  012644  004737  033274                         JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 4305  012650  154130                                 .WORD   154130  ;TASK DESCRIPTOR
 4306  012652  000404                                 BR      30$                     ;GO TO 30$ IF NO ERROR
 4307  012654  000240                                 NOP                     ;RETURN HERE IF ERROR
 4308  012656  104000                                 ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 4309  012660  000137  013452                         JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
 4310  012654                                  30$:
 4311
 4312                                          ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
 4313  012664  012737  000005  001376                 MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
 4314  012672  012702  001533                         MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
 4315  012676  112722  000006                         MOVB    #RMDA,(R2)+
 4316  012702  112722  000034                         MOVB    #RMDC,(R2)+
 4317  012706  112722  000032                         MOVB    #RMOF,(R2)+
 4318  012712  112722  000000                         MOVB    #RMCS1,(R2)+
 4319  012716  112722  000200                         MOVB    #200,(R2)+
 4320  012722  004737  037262                         JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
 4321  012726  000404                                 BR      40$     ;GO TO 40$ IF NO ERROR
 4322  012730  000240                                 NOP                             ;RETURN HERE IF ERROR
 4323  012732  104000                                 ERROR                           ;ERROR DEFINED BY PUT SUB
 4324  012734  000137  013452                         JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
 4325  012740                                  40$:
 4326
 4327                                          ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
 4328  012740  004737  036726                         JSR     PC,GETSTS               ;SETUP FOR STATUS
 4329  012744  004737  037622                         JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
 4330  012750                                  50$:
 4331
 4332                                          ;GO READ SEEK STATUS
```

```
 4333  012750  004737  037012              JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 4334  012754  000404                      BR      60$     ;GO TO 60$ IF NO ERROR
 4335  012756  000240                      NOP                     ;RETURN HERE IF ERROR
 4336  012760  104000                      ERROR                   ;ERROR DEFINED BY GET SUB
 4337  012762  000137  013452              JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
 4338  012766                      60$:
 4339
 4340                              ;VERIFY THE RESULTS OF THE SEEK COMMAND
 4341  012766  004737  044714              JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
 4342  012772  000405                      BR      70$                     ;GO TO 70$ IF NO ERROR
 4343  012774  000240                      NOP                     ;RETURN HERE IF ERROR
 4344  012776  1040C0                      ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
 4345  013000  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 4346  013002  000137  013452              JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
 4347  013006                      70$:
 4348
 4349                              ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
 4350  013006  012737  000063  001376      MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
```

```
 4351  013014  012702  001536                MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
 4352  013020  112722  000002                MOVB    #RMWC,(R2)+
 4353  013024  112722  000004                MOVB    #RMBA,(R2)+
 4354  013030  112722  000000                MOVB    #RMCS1,(R2)+
 4355  013034  112722  000200                MOVB    #200,(R2)+
 4356  013040  004737  037262                JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
 4357  013044  000404                        BR      80$     ;GO TO 80$ IF NO ERROR
 4358  013046  000240                        NOP                             ;RETURN HERE IF ERROR
 4359  013050  104000                        ERROR                           ;ERROR DEFINED BY PUT SUB
 4360  013052  000137  013452                JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
 4361  013056                        80$:
 4362
 4363                                ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
 4364  013056  004737  037622                JSR     PC,TIMOUT                       ;WAIT FOR COMMAND TO COMPLETE
 4365  013062  004737  037012                JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
 4366  013066  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
 4367  013070  000240                        NOP                             ;RETURN HERE IF ERROR
 4368  013072  104000                        ERROR                           ;ERROR DEFINED BY GET SUB
 4369  013074  000137  013452                JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
 4370  013100                        90$:
 4371
 4372                                ;VERIFY RESULTS OF WRITE COMMAND
 4373  013100  004737  040006                JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
 4374  013104  000405                        BR      100$                            ;GO TO 100$ IF NO ERROR
 4375  013106  000240                        NOP                             ;RETURN HERE IF ERROR
 4376  013110  104000                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
 4377  013112  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
 4378  013114  000137  013452                JMP     260$                            ;GO TO 260$ IF ERROR WAS FOUND
 4379  013120                        100$:
 4380  013120  004737  052312                JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
 4381  013124  000405                        BR      110$                            ;GO TO 110$ IF NO ERROR
 4382  013126  000240                        NOP                             ;RETURN HERE IF ERROR
 4383  013130  104000                        ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
 4384  013132  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
 4385  013134  000137  013452                JMP     260$                            ;GO TO 260$ IF ERROR WAS FOUND
 4386  013140                        110$:
 4387  013140  004737  040640                JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
 4388  013144  000405                        BR      120$                            ;GO TO 120$ IF NO ERROR
 4389  013146  000240                        NOP                             ;RETURN HERE IF ERROR
 4390  013150  104000                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
 4391  013152  004736                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
 4392  013154  000137  013452                JMP     260$                            ;GO TO 260$ IF ERROR WAS FOUND
 4393  013160                        120$:
 4394
 4395                                ;ALTER DATA BUFFER
 4396  013160  005137  103576                COM     BUFTWO-2                        ;COMPLEMENT LAST DATA WORD
 4397
 4398
 4399                                ;SETUP AND WRITE CHECK HEADER AND DATA COMMAND
 4400  013164  012737  000053  001376        MOV     #WCH!GO,RMCS10                  ;LOAD COMMAND
 4401  013172  004737  037262                JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
 4402  013176  000404                        BR      180$    ;GO TO 180$ IF NO ERROR
 4403  013200  000240                        NOP                             ;RETURN HERE IF ERROR
 4404  013202  104000                        ERROR                           ;ERROR DEFINED BY PUT SUB
 4405  013204  000137  013452                JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
 4406  013210                        180$:
```

```
4407
4408                                    ;WAIT FOR WRITE CHECK COMMAND TO COMPLETE AND READ ERRORS
4409  013210  004737  037622                    JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
4410  013214  004737  037012                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4411  013220  000404                            BR      190$    ;GO TO 190$ IF NO ERROR
4412  013222  000240                            NOP                             ;RETURN HERE IF ERROR
4413  013224  104000                            ERROR                           ;ERROR DEFINED BY GET SUB
4414  013226  000137  013452                    JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4415  013232                                190$:
4416
4417                                    ;CHECK FOR PRIMARY ERRORS
4418  013232  004737  040006                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4419  013236  000405                            BR      200$                    ;GO TO 200$ IF NO ERROR
4420  013240  000240                            NOP                             ;RETURN HERE IF ERROR
4421  013242  104000                            ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
4422  013244  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4423  013246  000137  013452                    JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4424  013252                                200$:
4425
```

```
 4426                                   ;MAKE SURE THE WRITE CHECK ERROR WAS DETECTED
 4427  013252  032737  040000  001336           BIT     #WCE,RMCS2I             ;IS WRITE CHECK ERROR SET??
 4428  013260  001023                           BNE     210$                    ;YES!!
 4429  013262  004737  052312                   JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 4430  013266  000405                           BR      205$                    ;GO TO 205$ IF NO ERROR
 4431  013270  000240                           NOP                     ;RETURN HERE IF ERROR
 4432  013272  104000                           ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 4433  013274  004736                           JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 4434  013276  000137  013452                   JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
 4435  013302  013737  001336  001140  205$:    MOV     RMCS2I,$GDDAT           ;LOAD EXPECTED STATUS
 4436  013310  052737  040000  001140           BIS     #WCE,$GDDAT
 4437  013316  013737  001336  001142           MOV     RMCS2I,$BDDAT           ;LOAD RECEIVED STATUS
 4438  013324  104337                           ERROR   337                     ;WRITE CHECK ERROR NOT DETECTED
 4439  013326  000451                           BR      260$
 4440  013330                         210$:
 4441
 4442                                   ;VERIFY THE ADDRESS OF THE WRITE CHECK ERROR
 4443  013330  012737  103576  001134           MOV     #BUFTWO-2,$GDADR        ;LOAD EXPECTED ADDRESS
 4444  013336  013737  001332  001136           MOV     RMBAI,$BDADR            ;LOAD RECEIVED ADDRESS
 4445  013344  162737  000002  001136           SUB     #2,$BDADR               ;DECREMENT RECEIVED ADDRESS
 4446
 4447                                   ;GET WCE DATA AND VERIFY IT IS OK
 4448  013352  112737  000022  001504           MOVB    #RMDB,GETINX            ;SETUP FOR READING RMDB
 4449  013360  112737  000200  001505           MOVB    #200,GETINX+1
 4450  013366  004737  037012                   JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 4451  013372  000404                           BR      230$    ;GO TO 230$ IF NO ERROR
 4452  013374  000240                           NOP                     ;RETURN HERE IF ERROR
 4453  013376  104000                           ERROR                   ;ERROR DEFINED BY GET SUB
 4454  013400  000137  013452                   JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
 4455  013404  013737  001350  001142  230$:    MOV     RMDBI,$BDDAT            ;LOAD RECEIVED DATA WORD
 4456  013412  013737  103576  001140           MOV     BUFTWO-2,$GDDAT         ;LOAD EXPECTED DATA WORD
 4457  013420  005137  001140                   COM     $GDDAT
 4458  013424  023737  001134  001136           CMP     $GDADR,$BDADR           ;IS ADDRESS OK??
 4459  013432  001402                           BEQ     220$                    ;YES!!
 4460  013434  104340                           ERROR   340                     ;ADDRESS OF WCE INCORRECT
 4461  013436  000405                           BR      260$
 4462  013440  023737  001140  001142  220$:    CMP     $GDDAT,$BDDAT           ;IS DATA WORD OK??
 4463  013446  001401                           BEQ     260$                    ;YES!!
 4464  013450  104341                           ERROR   341                     ;UNEXPECTED WCE DATA
 4465  013452                         260$:
 4466
 4467                                   ;;*****************************************************************
 4468                                   ;*TEST 11       FORMAT ONES - 16
 4469                                   ;;*****************************************************************
 4470  013452                         TST11:
 4471  013452  000004                           SCOPE                   ;SCOPE CALL
 4472  013454  000240                           NOP                     ;START OF TEST
 4473  013456  012706  001100                   MOV     #STACK,SP       ;INITIALIZE STACK POINTER
 4474  013462  013700  001276                   MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 4475  013466  013701  001446                   MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 4476  013472  012737  000011  001226           MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 4477  013500                         10$:
 4478
 4479                                   ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
 4480  013500  012737  000000  001432           MOV     #0,RMDC0                ;CYLINDER = 0
 4481  013506  012737  000000  001404           MOV     #0,RMDA0                ;TRACK = SECTOR = 0
```

```
 4482  013514  012737  010000  001430          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
 4483  013522  012737  177376  001400          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
 4484  013530  012737  102574  001402          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
 4485  013536  012737  000062  001376          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
 4486  013544  012737  064046  001174          MOV     #ONES,$TMP0             ;USE ALL ONES DATA PATTERN
 4487  013552  012737  000001  001176          MOV     #1,$TMP1
 4488  013560  004737  036114                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
 4489  013564                                20$:
 4490
 4491                                        ;PREPARE DEVICE FOR DATA TRANSFER
 4492  013564  004737  033274                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 4493  013570  154130                        .WORD   154130  ;TASK DESCRIPTOR
 4494  013572  000404                        BR      30$                     ;GO TO 30$ IF NO ERROR
 4495  013574  000240                        NOP                     ;RETURN HERE IF ERROR
 4496  013576  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 4497  013600  000137  014252                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 4498  013604                                30$:
 4499
 4500                                        ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
 4501  013604  012737  000005  001376          MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
 4502  013612  012702  001533                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
 4503  013616  112722  000006                  MOVB    #RMDA,(R2)+
 4504  013622  112722  000034                  MOVB    #RMDC,(R2)+
 4505  013626  112722  000032                  MOVB    #RMOF,(R2)+
 4506  013632  112722  000000                  MOVB    #RMCS1,(R2)+
 4507  013636  112722  000200                  MOVB    #200,(R2)+
 4508  013642  004737  037262                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 4509  013646  000404                        BR      40$     ;GO TO 40$ IF NO ERROR
 4510  013650  000240                        NOP                     ;RETURN HERE IF ERROR
 4511  013652  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
 4512  013654  000137  014252                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 4513  013660                                40$:
 4514
 4515                                        ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
 4516  013660  004737  036726                JSR     PC,GETSTS               ;SETUP FOR STATUS
 4517  013664  004737  037622                JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
 4518  013670                                50$:
 4519
 4520                                        ;GO READ SEEK STATUS
 4521  013670  004737  037012                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 4522  013674  000404                        BR      60$     ;GO TO 60$ IF NO ERROR
 4523  013676  000240                        NOP                     ;RETURN HERE IF ERROR
 4524  013700  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
 4525  013702  000137  014252                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 4526  013706                                60$:
 4527
 4528                                        ;VERIFY THE RESULTS OF THE SEEK COMMAND
 4529  013706  004737  044714                JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
 4530  013712  000405                        BR      70$                     ;GO TO 70$ IF NO ERROR
 4531  013714  000240                        NOP                     ;RETURN HERE IF ERROR
 4532  013716  104000                        ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
 4533  013720  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 4534  013722  000137  014252                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 4535  013726                                70$:
 4536
 4537                                        ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
```

```
4538  013726  012737  000063  001376          MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
4539  013734  012702  001536                  MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
4540  013740  112722  000002                  MOVB    #RMWC,(R2)+
4541  013744  112722  000004                  MOVB    #RMBA,(R2)+
4542  013750  112722  000000                  MOVB    #RMCS1,(R2)+
4543  013754  112722  000200                  MOVB    #200,(R2)+
4544  013760  004737  037262                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
4545  013764  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
4546  013766  000240                          NOP                             ;RETURN HERE IF ERROR
4547  013770  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
4548  013772  000137  014252                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4549  013776                          80$:
4550
4551                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4552  013776  004737  037622                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
4553  014002  004737  037012                  JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
4554  014006  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
4555  014010  000240                          NOP                             ;RETURN HERE IF ERROR
4556  014012  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
4557  014014  000137  014252                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4558  014020                          90$:
4559
4560                                  ;VERIFY RESULTS OF WRITE COMMAND
4561  014020  004737  040006                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
4562  014024  000405                          BR      100$                            ;GO TO 100$ IF NO ERROR
4563  014026  000240                          NOP                             ;RETURN HERE IF ERROR
4564  014030  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
4565  014032  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4566  014034  000137  014252                  JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
4567  014040                          100$:
4568  014040  004737  052312                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
4569  014044  000405                          BR      110$                            ;GO TO 110$ IF NO ERROR
4570  014046  000240                          NOP                             ;RETURN HERE IF ERROR
4571  014050  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
4572  014052  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4573  014054  000137  014252                  JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
4574  014060                          110$:
4575  014060  004737  040640                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
4576  014064  000405                          BR      120$                            ;GO TO 120$ IF NO ERROR
4577  014066  000240                          NOP                             ;RETURN HERE IF ERROR
4578  014070  104000                          ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
4579  014072  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4580  014074  000137  014252                  JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
4581  014100                          120$:
4582
4583
4584                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
4585  014100  012737  000073  001376          MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
4586  014106  012737  103600  001402          MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
4587  014114  004737  037262                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
4588  014120  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
4589  014122  000240                          NOP                             ;RETURN HERE IF ERROR
4590  014124  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
4591  014126  000137  014252                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4592  014132                          130$:
4593
```

```
4594
4595  014132  004737  037622          ;WAIT FOR READ TO COMPLETE AND GET STATUS
4596  014136  004737  037012                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
4597  014142  000404                          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4598  014144  000240                          BR      140$    ;GO TO 140$ IF NO ERROR
4599  014146  104000                          NOP                     ;RETURN HERE IF ERROR
4600  014150  000137  014252                  ERROR                   ;ERROR DEFINED BY GET SUB
4601  014154                                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
4602                                  140$:
4603
4604  014154  004737  040006          ;VERIFY THE RESULTS OF READ OPERATION
4605  014160  000405                          JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
4606  014162  000240                          BR      150$                    ;GO TO 150$ IF NO ERROR
4607  014164  104000                          NOP                     ;RETURN HERE IF ERROR
4608  014166  004736                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4609  014170  000137  014252                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4610  014174                                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4611  014174  004737  052312          150$:
4612  014200  000405                          JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
4613  014202  000240                          BR      160$                    ;GO TO 160$ IF NO ERROR
4614  014204  104000                          NOP                     ;RETURN HERE IF ERROR
4615  014206  004736                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4616  014210  000137  014252                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4617  014214                                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4618  014214  004737  040640          160$:
4619  014220  000405                          JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
4620  014222  000240                          BR      170$                    ;GO TO 170$ IF NO ERROR
4621  014224  104000                          NOP                     ;RETURN HERE IF ERROR
4622  014226  004736                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4623  014230  000137  014252                  JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4624  014234                                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
4625                                  170$:
4626                                  ;VERIFY DATA
4627  014234  004737  036360                  JSR     PC,CMPBUF               ;GO COMPARE WRITE, READ DATA BUFFERS
4628  014240  102574                          .WORD   BUFONE                  ;STARTING ADDRESS OF WRITE BUFFER
4629  014242  103600                          .WORD   BUFTWO                  ;STARTING ADDDRESS OF READ BUFFER
4630  014244  000402                          BR      180$                    ;GO TO 180$ IF NO ERROR
4631  014246  000240                          NOP                     ;RETURN HERE IF ERROR
4632  014250  104000                          ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
4633  014252                                  180$:
4634
4635                                  ;;*****************************************************************
4636                                  ;*TEST 12       FORMAT CHECK ONES - 16
4637                                  ;*****************************************************************
4638  014252                          TST12:
4639  014252  000004                          SCOPE                   ;SCOPE CALL
4640  014254  000240                          NOP                     ;START OF TEST
4641  014256  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
4642  014262  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
4643  014266  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
4644  014272  012737  000012  001226          MOV     #12,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4645  014300                                  10$:
4646
4647                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
4648  014300  012737  000000  001432          MOV     #0,RMDC0                ;CYLINDER = 0
4649  014306  012737  000000  001404          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
```

```
4650  014314  012737  010000  001430          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
4651  014322  012737  177376  001400          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
4652  014330  012737  102574  001402          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
4653  014336  012737  000062  001376          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
4654  014344  012737  064046  001174          MOV     #ONES,$TMP0             ;USE ALL ONES DATA PATTERN
4655  014352  012737  000001  001176          MOV     #1,$TMP1
4656  014360  004737  036114                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
4657  014364                                  20$:
4658
4659                                          ;PREPARE DEVICE FOR DATA TRANSFER
4660  014364  004737  033274                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
4661  014370  154130                          .WORD   154130  ;TASK DESCRIPTOR
4662  014372  000404                          BR      30$             ;GO TO 30$ IF NO ERROR
4663  014374  000240                          NOP                     ;RETURN HERE IF ERROR
4664  014376  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
4665  014400  000137  015026                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4666  014404                                  30$:
4667
4668                                          ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
4669  014404  012737  000005  001376          MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
4670  014412  012702  001533                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
4671  014416  112722  000006                  MOVB    #RMDA,(R2)+
4672  014422  112722  000034                  MOVB    #RMDC,(R2)+
4673  014426  112722  000032                  MOVB    #RMOF,(R2)+
4674  014432  112722  000000                  MOVB    #RMCS1,(R2)+
4675  014436  112722  000200                  MOVB    #200,(R2)+
4676  014442  004737  037262                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4677  014446  000404                          BR      40$     ;GO TO 40$ IF NO ERROR
4678  014450  000240                          NOP                     ;RETURN HERE IF ERROR
4679  014452  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
4680  014454  000137  015026                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4681  014460                                  40$:
4682
4683                                          ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
4684  014460  004737  036726                  JSR     PC,GETSTS               ;SETUP FOR STATUS
4685  014464  004737  037622                  JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
4686  014470                                  50$:
4687
4688                                          ;GO READ SEEK STATUS
4689  014470  004737  037012                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4690  014474  000404                          BR      60$     ;GO TO 60$ IF NO ERROR
4691  014476  000240                          NOP                     ;RETURN HERE IF ERROR
4692  014500  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4693  014502  000137  015026                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4694  014506                                  60$:
4695
4696                                          ;VERIFY THE RESULTS OF THE SEEK COMMAND
4697  014506  004737  044714                  JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
4698  014512  000405                          BR      70$                     ;GO TO 70$ IF NO ERROR
4699  014514  000240                          NOP                     ;RETURN HERE IF ERROR
4700  014516  104000                          ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
4701  014520  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4702  014522  000137  015026                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4703  014526                                  70$:
4704
4705                                          ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
```

```
4706  014526  012737  000063  001376          MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
4707  014534  012702  001536                  MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
4708  014540  112722  000002                  MOVB    #RMWC,(R2)+
4709  014544  112722  000004                  MOVB    #RMBA,(R2)+
4710  014550  112722  000000                  MOVB    #RMCS1,(R2)+
4711  014554  112722  000200                  MOVB    #200,(R2)+
4712  014560  004737  037262                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4713  014564  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
4714  014566  000240                          NOP                     ;RETURN HERE IF ERROR
4715  014570  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
4716  014572  000137  015026                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4717  014576                          80$:
4718
4719                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
4720  014576  004737  037622                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
4721  014602  004737  037012                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4722  014606  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
4723  014610  000240                          NOP                     ;RETURN HERE IF ERROR
4724  014612  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
4725  014614  000137  015026                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4726  014620                          90$:
4727
4728                                  ;VERIFY RESULTS OF WRITE COMMAND
4729  014620  004737  040006                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4730  014624  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
4731  014626  000240                          NOP                     ;RETURN HERE IF ERROR
4732  014630  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4733  014632  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4734  014634  000137  015026                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4735  014640                          100$:
4736  014640  004737  052312                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4737  014644  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
4738  014646  000240                          NOP                     ;RETURN HERE IF ERROR
4739  014650  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4740  014652  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4741  014654  000137  015026                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4742  014660                          110$:
4743  014660  004737  040640                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
4744  014664  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
4745  014666  000240                          NOP                     ;RETURN HERE IF ERROR
4746  014670  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
4747  014672  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4748  014674  000137  015026                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4749  014700                          120$:
4750
4751
4752                                  ;WRITE CHECK HEADER AND DATA FOR SECTOR JUST WRITTEN
4753  014700  012737  000053  001376          MOV     #WCH!GO,RMCS10          ;WRITE CHECK COMMAND
4754  014706  004737  037262                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
4755  014712  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
4756  014714  000240                          NOP                     ;RETURN HERE IF ERROR
4757  014716  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
4758  014720  000137  015026                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4759  014724                          130$:
4760
4761                                  ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
```

```
4762  014724  004737  037622                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
4763  014730  004737  037012                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4764  014734  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
4765  014736  000240                          NOP                             ;RETURN HERE IF ERROR
4766  014740  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
4767  014742  000137  015026                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4768  014746                          140$:
4769
4770                                  ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
4771  014746  004737  040006                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
4772  014752  000405                          BR      150$                            ;GO TO 150$ IF NO ERROR
4773  014754  000240                          NOP                             ;RETURN HERE IF ERROR
4774  014756  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
4775  014760  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4776  014762  000137  015026                  JMP     260$                            ;GO TO 260$ IF ERROR WAS FOUND
4777  014766                          150$:
4778  014766  004737  052312                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
4779  014772  000405                          BR      160$                            ;GO TO 160$ IF NO ERROR
4780  014774  000240                          NOP                             ;RETURN HERE IF ERROR
4781  014776  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
4782  015000  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4783  015002  000137  015026                  JMP     260$                            ;GO TO 260$ IF ERROR WAS FOUND
4784  015006                          160$:
4785  015006  004737  040640                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
4786  015012  000405                          BR      170$                            ;GO TO 170$ IF NO ERROR
4787  015014  000240                          NOP                             ;RETURN HERE IF ERROR
4788  015016  104000                          ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
4789  015020  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
4790  015022  000137  015026                  JMP     260$                            ;GO TO 260$ IF ERROR WAS FOUND
4791  015026                          170$:
4792
4793  015026                          260$:
4794
4795                                  ;;*****************************************************************
4796                                  ;*TEST 13       FORMAT CHECK ONES W/ WCE ERRORS
4797                                  ;;*****************************************************************
4798  015026                          TST13:
4799  015026  000004                          SCOPE                           ;SCOPE CALL
4800  015030  000240                          NOP                             ;START OF TEST
4801  015032  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
4802  015036  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
4803  015042  013701  001446                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
4804  015046  012737  000013  001226          MOV     #13,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
4805  015054                          10$:
4806
4807                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
4808  015054  012737  000000  001432          MOV     #0,RMDC0                        ;CYLINDER = 0
4809  015062  012737  000000  001404          MOV     #0,RMDA0                        ;TRACK = SECTOR = 0
4810  015070  012737  010000  001430          MOV     #FMT16,RMOF0                    ;16 BIT FORMAT
4811  015076  012737  177376  001400          MOV     #<^C<2+256.>+1>,RMWC0           ;2 + 256 WORDS
4812  015104  012737  102574  001402          MOV     #BUFONE,RMBA0                   ;DATA BUFFER ADDRESS
4813  015112  012737  000062  001376          MOV     #WH,RMCS10                      ;WRITE HEADER AND DATA
4814  015120  012737  064046  001174          MOV     #ONES,$TMP0                     ;USE ALL ONES DATA PATTERN
4815  015126  012737  000001  001176          MOV     #1,$TMP1
4816  015134  004737  036114                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
4817  015140                          20$:
```

```
                                        ;PREPARE DEVICE FOR DATA TRANSFER
015140  004737  033274                          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
015144  154130                                  .WORD   154130  ;TASK DESCRIPTOR
015146  000404                                  BR      30$             ;GO TO 30$ IF NO ERROR
015150  000240                                  NOP                     ;RETURN HERE IF ERROR
015152  104000                                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
015154  000137  015744                          JMP     260$            ;GO TO 260$ IF ERROR WAS FOUND
015160                                  30$:

                                        ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
015160  012737  000005  001376                  MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
015166  012702  001533                          MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
015172  112722  000006                          MOVB    #RMDA,(R2)+
015176  112722  000034                          MOVB    #RMDC,(R2)+
015202  112722  000032                          MOVB    #RMOF,(R2)+
015206  112722  000000                          MOVB    #RMCS1,(R2)+
015212  112722  000200                          MOVB    #200,(R2)+
015216  004737  037262                          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
015222  000404                                  BR      40$     ;GO TO 40$ IF NO ERROR
015224  000240                                  NOP                     ;RETURN HERE IF ERROR
015226  104000                                  ERROR                   ;ERROR DEFINED BY PUT SUB
015230  000137  015744                          JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
015234                                  40$:

                                        ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
015234  004737  036726                          JSR     PC,GETSTS               ;SETUP FOR STATUS
015240  004737  037622                          JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
015244                                  50$:

                                        ;GO READ SEEK STATUS
015244  004737  037012                          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
015250  000404                                  BR      60$     ;GO TO 60$ IF NO ERROR
015252  000240                                  NOP                     ;RETURN HERE IF ERROR
015254  104000                                  ERROR                   ;ERROR DEFINED BY GET SUB
015256  000137  015744                          JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
015262                                  60$:

                                        ;VERIFY THE RESULTS OF THE SEEK COMMAND
015262  004737  044714                          JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
015266  000405                                  BR      70$                     ;GO TO 70$ IF NO ERROR
015270  000240                                  NOP                     ;RETURN HERE IF ERROR
015272  104000                                  ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
015274  004736                                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
015276  000137  015744                          JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
015302                                  70$:

                                        ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
015302  012737  000063  001376                  MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
015310  012702  001536                          MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
015314  112722  000002                          MOVB    #RMWC,(R2)+
015320  112722  000004                          MOVB    #RMBA,(R2)+
015324  112722  000000                          MOVB    #RMCS1,(R2)+
015330  112722  000200                          MOVB    #200,(R2)+
015334  004737  037262                          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
015340  000404                                  BR      80$     ;GO TO 80$ IF NO ERROR
```

```
 4874  015342  000240                          NOP                     ;RETURN HERE IF ERROR
 4875  015344  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
 4876  015346  000137  015744                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
 4877  015352                          80$:
 4878
 4879                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
 4880  015352  004737  037622                  JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
 4881  015356  004737  037012                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 4882  015362  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
 4883  015364  000240                          NOP                     ;RETURN HERE IF ERROR
 4884  015366  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
 4885  015370  000137  015744                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
 4886  015374                          90$:
 4887
 4888                                  ;VERIFY RESULTS OF WRITE COMMAND
 4889  015374  004737  040006                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 4890  015400  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
 4891  015402  000240                          NOP                     ;RETURN HERE IF ERROR
 4892  015404  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 4893  015406  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 4894  015410  000137  015744                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
 4895  015414                          100$:
 4896  015414  004737  052312                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 4897  015420  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
 4898  015422  000240                          NOP                     ;RETURN HERE IF ERROR
 4899  015424  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 4900  015426  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 4901  015430  000137  015744                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
 4902  015434                          110$:
 4903  015434  004737  040640                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 4904  015440  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
 4905  015442  000240                          NOP                     ;RETURN HERE IF ERROR
 4906  015444  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 4907  015446  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 4908  015450  000137  015744                  JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
 4909  015454                          120$:
 4910
 4911                                  ;ALTER DATA BUFFER
 4912  015454  005137  103576                  COM     BUFTWO-2                ;COMPLEMENT DATA WORD
 4913
 4914
 4915                                  ;SETUP AND WRITE CHECK HEADER AND DATA COMMAND
 4916  015460  012737  000053  001376          MOV     #WCH!GO,RMCS10          ;LOAD COMMAND
 4917  015466  004737  037262                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 4918  015472  000404                          BR      180$    ;GO TO 180$ IF NO ERROR
 4919  015474  000240                          NOP                     ;RETURN HERE IF ERROR
 4920  015476  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
 4921  015500  000137  015744                  JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
 4922  015504                          180$:
 4923
 4924                                  ;WAIT FOR WRITE CHECK COMMAND TO COMPLETE AND READ STATUS
 4925  015504  004737  037622                  JSR     PC,TIMOUT               ;WAIT FOR COMMMAND TO COMPLETE
 4926  015510  004737  037012                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 4927  015514  000404                          BR      190$    ;GO TO 190$ IF NO ERROR
 4928  015516  000240                          NOP                     ;RETURN HERE IF ERROR
 4929  015520  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
```

```
4930  015522  000137  015744                JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4931  015526                        190$:
4932
4933                                ;CHECK FOR PRIMARY ERRORS
4934  015526  004737  040006                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
4935  015532  000405                        BR      200$                    ;GO TO 200$ IF NO ERROR
4936  015534  000240                        NOP                     ;RETURN HERE IF ERROR
4937  015536  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
4938  015540  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4939  015542  000137  015744                JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4940  015546                        200$:
4941
4942                                ;MAKE SURE THE WRITE CHECK ERROR WAS DETECTED
4943  015546  032737  040000  001336        BIT     #WCE,RMCS2I             ;IS WRITE CHECK ERROR SET??
4944  015554  001022                        BNE     210$                    ;YES!!
4945  015556  004737  052312                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
4946  015562  000405                        BR      205$                    ;GO TO 205$ IF NO ERROR
4947  015564  000240                        NOP                     ;RETURN HERE IF ERROR
4948  015566  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
4949  015570  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
4950  015572  000137  015744                JMP     260$                    ;GO TO 260$ IF ERROR WAS FOUND
4951  015576  013737  001336  001140  205$:  MOV     RMCS2I,$GDDAT           ;LOAD EXPECTED STATUS
4952  015604  052737  040000  001140        BIS     #WCE,$GDDAT
4953  015612  013737  001336  001142        MOV     RMCS2I,$BDDAT           ;LOAD RECEIVED STATUS
4954  015620  104337                        ERROR   337                     ;WRITE CHECK ERROR NOT DETECTED
4955  015622                        210$:
4956
4957                                ;VERIFY THE ADDRESS OF THE WRITE CHECK ERROR
4958  015622  012737  103576  001134        MOV     #BUFTWO-2,$GDADR        ;LOAD EXPECTED ADDRESS
4959  015630  013737  001332  001136        MOV     RMBAI,$BDADR            ;LOAD RECEIVED ADDRESS
4960  015636  162737  000002  001136        SUB     #2,$BDADR               ;DECREMENT RECEIVED ADDRESS
4961
4962                                ;GET WCE DATA AND VERIFY IT IS OK
4963  015644  112737  000022  001504        MOVB    #RMDB,GETINX            ;SETUP FOR READING RMDB
4964  015652  112737  000200  001505        MOVB    #200,GETINX+1
4965  015660  004737  037012                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
4966  015664  000404                        BR      230$    ;GO TO 230$ IF NO ERROR
4967  015666  000240                        NOP                     ;RETURN HERE IF ERROR
4968  015670  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
4969  015672  000137  015744                JMP     260$    ;GO TO 260$ IF ERROR WAS FOUND
4970  015676  013737  001350  001142  230$:  MOV     RMDBI,$BDDAT            ;LOAD RECEIVED DATA WORD
4971  015704  013737  103576  001140        MOV     BUFTWO-2,$GDDAT         ;LOAD EXPECTED DATA WORD
4972  015712  005137  001140                COM     $GDDAT
4973  015716  023737  001134  001136        CMP     $GDADR,$BDADR           ;IS ADDRESS OK??
4974  015724  001402                        BEQ     220$                    ;YES!!
4975  015726  104340                        ERROR   340                     ;ADDRESS OF WCE INCORRECT
4976  015730  000405                        BR      260$
4977  015732                        220$:
4978  015732  023737  001140  001142        CMP     $GDDAT,$BDDAT           ;IS DATA WORD OK??
4979  015740  001401                        BEQ     260$                    ;YES!!
4980  015742  104341                        ERROR   341                     ;UNEXPECTED WCE DATA
4981  015744                        260$:
4982
4983                                ;;*****************************************************************
4984                                ;*TEST 14       FORMAT MULTIPLE SECTORS
4985                                ;;*****************************************************************
```

```
 4986  015744
 4987  015744  000004                        TST14:
 4988  015746  000240                                SCOPE                   ;SCOPE CALL
 4989  015750  012706  001100                        NOP                     ;START OF TEST
 4990  015754  013700  001276                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
 4991  015760  013701  001446                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 4992  015764  012737  000014  001226                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 4993  015772                                        MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 4994                                        10$:
 4995                                        ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
 4996  015772  012737  000000  001432                MOV     #0,RMDC0                ;CYLINDER = 0
 4997  016000  012737  000000  001404                MOV     #0,RMDA0                ;TRACK = SECTOR = 0
 4998  016006  012737  010000  001430                MOV     #FMT16,RMOF0            ;16 BIT FORMAT
 4999  016014  012737  176774  001400                MOV     #<^C<<2+256.>*2>+1>,RMWC0;WORD COUNT FOR 2 SECTORS
 5000  016022  012737  102574  001402                MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
 5001  016030  012737  000062  001376                MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
 5002  016036  012737  064110  001174                MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
 5003  016044  012737  000001  001176                MOV     #1,$TMP1
 5004  016052  004737  036114                        JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
 5005  016056                                20$:
 5006
 5007                                        ;PREPARE DEVICE FOR DATA TRANSFER
 5008  016056  004737  033274                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 5009  016062  154130                                .WORD   154130  ;TASK DESCRIPTOR
 5010  016064  000404                                BR      30$             ;GO TO 30$ IF NO ERROR
 5011  016066  000240                                NOP                     ;RETURN HERE IF ERROR
 5012  016070  104000                                ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 5013  016072  000137  016520                        JMP     180$            ;GO TO 180$ IF ERROR WAS FOUND
 5014  016076                                30$:
 5015
 5016                                        ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
 5017  016076  012737  000005  001376                MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
 5018  016104  012702  001533                        MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
 5019  016110  112722  000006                        MOVB    #RMOA,(R2)+
 5020  016114  112722  000034                        MOVB    #RMDC,(R2)+
 5021  016120  112722  000032                        MOVB    #RMOF,(R2)+
 5022  016124  112722  000000                        MOVB    #RMCS1,(R2)+
 5023  016130  112722  000200                        MOVB    #200,(R2)+
 5024  016134  004737  037262                        JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 5025  016140  000404                                BR      40$     ;GO TO 40$ IF NO ERROR
 5026  016142  000240                                NOP                     ;RETURN HERE IF ERROR
 5027  016144  104000                                ERROR                   ;ERROR DEFINED BY PUT SUB
 5028  016146  000137  016520                        JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 5029  016152                                40$:
 5030
 5031                                        ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
 5032  016152  004737  036726                        JSR     PC,GETSTS               ;SETUP FOR STATUS
 5033  016156  004737  037622                        JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
 5034  016162                                50$:
 5035
 5036                                        ;GO READ SEEK STATUS
 5037  016162  004737  037012                        JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 5038  016166  000404                                BR      60$     ;GO TO 60$ IF NO ERROR
 5039  016170  000240                                NOP                     ;RETURN HERE IF ERROR
 5040  016172  104000                                ERROR                   ;ERROR DEFINED BY GET SUB
 5041  016174  000137  016520                        JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
```

```
5042  016200                                  60$:
5043
5044                                          ;VERIFY THE RESULTS OF THE SEEK COMMAND
5045  016200  004737  044714                         JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
5046  016204  000405                                 BR      70$                     ;GO TO 70$ IF NO ERROR
5047  016206  000240                                 NOP                     ;RETURN HERE IF ERROR
5048  016210  104000                                 ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5049  016212  004736                                 JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5050  016214  000137  016520                         JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5051  016220                                  70$:
5052
5053                                          ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5054  016220  012737  000063  001376                 MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
5055  016226  012702  001536                         MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
5056  016232  112722  000002                         MOVB    #RMWC,(R2)+
5057  016236  112722  000004                         MOVB    #RMBA,(R2)+
5058  016242  112722  000000                         MOVB    #RMCS1,(R2)+
5059  016246  112722  000200                         MOVB    #200,(R2)+
5060  016252  004737  037262                         JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5061  016256  000404                                 BR      80$     ;GO TO 80$ IF NO ERROR
5062  016260  000240                                 NOP                     ;RETURN HERE IF ERROR
5063  016262  104000                                 ERROR                   ;ERROR DEFINED BY PUT SUB
5064  016264  000137  016520                         JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5065  016270                                  80$:
5066
5067                                          ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5068  016270  004737  037622                         JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
5069  016274  004737  037012                         JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5070  016300  000404                                 BR      90$     ;GO TO 90$ IF NO ERROR
5071  016302  000240                                 NOP                     ;RETURN HERE IF ERROR
5072  016304  104000                                 ERROR                   ;ERROR DEFINED BY GET SUB
5073  015306  000137  016520                         JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5074  016312                                  90$:
5075
5076                                          ;VERIFY RESULTS OF WRITE COMMAND
5077  016312  004737  040006                         JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5078  016316  000405                                 BR      100$                    ;GO TO 100$ IF NO ERROR
5079  016320  000240                                 NOP                     ;RETURN HERE IF ERROR
5080  016322  104000                                 ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5081  016324  004736                                 JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5082  016326  000137  016520                         JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5083  016332                                  100$:
5084  016332  004737  052312                         JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5085  016336  000405                                 BR      110$                    ;GO TO 110$ IF NO ERROR
5086  016340  000240                                 NOP                     ;RETURN HERE IF ERROR
5087  016342  104000                                 ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5088  016344  004736                                 JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5089  016346  000137  016520                         JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5090  016352                                  110$:
5091  016352  004737  040640                         JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5092  016356  000405                                 BR      120$                    ;GO TO 120$ IF NO ERROR
5093  016360  000240                                 NOP                     ;RETURN HERE IF ERROR
5094  016362  104000                                 ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5095  016364  004736                                 JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5096  016366  000137  016520                         JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5097  016372                                  120$:
```

```
                                ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
016372  012737  000053  001376          MOV     #WCH!GO,RMCS10          ;READ HEADER & DATA COMMAND
016400  004737  037262                  JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
016404  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
016406  000240                          NOP                             ;RETURN HERE IF ERROR
016410  104000                          ERROR                           ;ERROR DEFINED BY PUT SUB
016412  000137  016520                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
016416                          130$:
                                ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
016416  004737  037622                  JSR     PC,TIMOUT                       ;WAIT FOR WRITE CHECK TO FINISH
016422  004737  037012                  JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
016426  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
016430  000240                          NOP                             ;RETURN HERE IF ERROR
016432  104000                          ERROR                           ;ERROR DEFINED BY GET SUB
016434  000137  016520                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
016440                          140$:
                                ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
016440  004737  040006                  JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
016444  000405                          BR      150$                            ;GO TO 150$ IF NO ERROR
016446  000240                          NOP                             ;RETURN HERE IF ERROR
016450  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
016452  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
016454  000137  016520                  JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
016460                          150$:
016460  004737  052312                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
016464  000405                          BR      160$                            ;GO TO 160$ IF NO ERROR
016466  000240                          NOP                             ;RETURN HERE IF ERROR
016470  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
016472  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
016474  000137  016520                  JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
016500                          160$:
016500  004737  040640                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
016504  000405                          BR      170$                            ;GO TO 170$ IF NO ERROR
016506  000240                          NOP                             ;RETURN HERE IF ERROR
016510  104000                          ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
016512  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
016514  000137  016520                  JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
016520                          170$:
016520                          180$:

                                ;;*****************************************************************
                                ;*TEST 15        FORMAT W/ HEAD SWITCHING
                                ;;*****************************************************************
016520                          TST15:
016520  000004                          SCOPE                           ;SCOPE CALL
016522  000240                          NOP                             ;START OF TEST
016524  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
016530  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
016534  013701  001446                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
016540  012737  000015  001226          MOV     #15,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
016546                          10$:
```

```
5154                                    ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5155  016546  012737  000000  001432            MOV     #0,RMDC0                ;CYLINDER = 0
5156  016554  012737  000037  001404            MOV     #31.,RMDA0              ;START AT LAST SECTOR
5157  016562  012737  010000  001430            MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5158  016570  012737  176774  001400            MOV     #(^C<(2+256.)*2)+1),RMWC0;WORD COUNT FOR 2 SECTORS
5159  016576  012737  102574  001402            MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
5160  016604  012737  000062  001376            MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
5161  016612  012737  064110  001174            MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
5162  016620  012737  000001  001176            MOV     #1,$TMP1
5163  016626  004737  036114                    JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
5164  016632                                  20$:
5165
5166                                    ;PREPARE DEVICE FOR DATA TRANSFER
5167  016632  004737  033274                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5168  016636  154130                            .WORD   154130  ;TASK DESCRIPTOR
5169  016640  000404                            BR      30$                     ;GO TO 30$ IF NO ERROR
5170  016642  000240                            NOP                     ;RETURN HERE IF ERROR
5171  016644  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5172  016646  000137  017274                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5173  016652                                  30$:
5174
5175                                    ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
5176  016652  012737  000005  001376            MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
5177  016660  012702  001533                    MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
5178  016664  112722  000006                    MOVB    #RMDA,(R2)+
5179  016670  112722  000034                    MOVB    #RMDC,(R2)+
5180  016674  112722  000032                    MOVB    #RMOF,(R2)+
5181  016700  112722  000000                    MOVB    #RMCS1,(R2)+
5182  016704  112722  000200                    MOVB    #200,(R2)+
5183  016710  004737  037262                    JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5184  016714  000404                            BR      40$     ;GO TO 40$ IF NO ERROR
5185  016716  000240                            NOP                     ;RETURN HERE IF ERROR
5186  016720  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
5187  016722  000137  017274                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5188  016726                                  40$:
5189
5190                                    ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
5191  016726  004737  036726                    JSR     PC,GETSTS               ;SETUP FOR STATUS
5192  016732  004737  037622                    JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
5193  016736                                  50$:
5194
5195                                    ;GO READ SEEK STATUS
5196  016736  004737  037012                    JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
5197  016742  000404                            BR      60$     ;GO TO 60$ IF NO ERROR
5198  016744  000240                            NOP                     ;RETURN HERE IF ERROR
5199  016746  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
5200  016750  000137  017274                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5201  016754                                  60$:
5202
5203                                    ;VERIFY THE RESULTS OF THE SEEK COMMAND
5204  016754  004737  044714                    JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
5205  016760  000405                            BR      70$                     ;GO TO 70$ IF NO ERROR
5206  016762  000240                            NOP                     ;RETURN HERE IF ERROR
5207  016764  104000                            ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5208  016766  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5209  016770  000137  017274                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
```

```
5210  016774                                  70$:
5211
5212                                          ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5213  016774  012737  000063  001376                MOV     #WH!GO,RMCS1O           ;LOAD WRITE HEADER AND DATA
5214  017002  012702  001536                        MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
5215  017006  112722  000002                        MOVB    #RMWC,(R2)+
5216  017012  112722  000004                        MOVB    #RMBA,(R2)+
5217  017016  112722  000000                        MOVB    #RMCS1,(R2)+
5218  017022  112722  000200                        MOVB    #200,(R2)+
5219  017026  004737  037262                        JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5220  017032  000404                                BR      80$     ;GO TO 80$ IF NO ERROR
5221  017034  000240                                NOP                             ;RETURN HERE IF ERROR
5222  017036  104000                                ERROR                           ;ERROR DEFINED BY PUT SUB
5223  017040  000137  017274                        JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5224  017044                                  80$:
5225
5226                                          ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5227  017044  004737  037622                        JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
5228  017050  004737  037012                        JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
5229  017054  000404                                BR      90$     ;GO TO 90$ IF NO ERROR
5230  017056  000240                                NOP                             ;RETURN HERE IF ERROR
5231  017060  104000                                ERROR                           ;ERROR DEFINED BY GET SUB
5232  017062  000137  017274                        JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5233  017066                                  90$:
5234
5235                                          ;VERIFY RESULTS OF WRITE COMMAND
5236  017[illegible]  004737  040006                JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
5237  017[illegible]  000405                        BR      100$                    ;GO TO 100$ IF NO ERROR
5238  017074  000240                                NOP                             ;RETURN HERE IF ERROR
5239  017076  104000                                ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
5240  017100  004736                                JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5241  017102  000137  017274                        JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5242  017106                                  100$:
5243  017106  004737  052312                        JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
5244  017112  000405                                BR      110$                    ;GO TO 110$ IF NO ERROR
5245  017114  000240                                NOP                             ;RETURN HERE IF ERROR
5246  017116  104000                                ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
5247  017120  004736                                JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5248  017122  000137  017274                        JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5249  017126                                  110$:
5250  017126  004737  040640                        JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
5251  017132  000405                                BR      120$                    ;GO TO 120$ IF NO ERROR
5252  017134  000240                                NOP                             ;RETURN HERE IF ERROR
5253  017136  104000                                ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
5254  017140  004736                                JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5255  017142  000137  017274                        JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5256  017146                                  120$:
5257
5258
5259                                          ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
5260  017146  012737  000053  001376                MOV     #WCH!GO,RMCS1O          ;READ HEADER & DATA COMMAND
5261  017154  004737  037262                        JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5262  017160  000404                                BR      130$    ;GO TO 130$ IF NO ERROR
5263  017162  000240                                NOP                             ;RETURN HERE IF ERROR
5264  017164  104000                                ERROR                           ;ERROR DEFINED BY PUT SUB
5265  017166  000137  017274                        JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
```

```
  5266  017172                                130$:
  5267
  5268                                        ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
  5269  017172  004737  037622                      JSR     PC,TIMOUT               ;WAIT FOR WRITE CHECK TO FINISH
  5270  017176  004737  037012                      JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
  5271  017202  000404                            BR      140$    ;GO TO 140$ IF NO ERROR
  5272  017204  000240                            NOP                     ;RETURN HERE IF ERROR
  5273  017206  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
  5274  017210  000137  017274                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
  5275  017214                                140$:
  5276
  5277                                        ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
  5278  017214  004737  040006                      JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
  5279  017220  000405                            BR      150$                    ;GO TO 150$ IF NO ERROR
  5280  017222  000240                            NOP                     ;RETURN HERE IF ERROR
  5281  017224  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
  5282  017226  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
  5283  017230  000137  017274                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
  5284  017234                                150$:
  5285  017234  004737  052312                      JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
  5286  017240  000405                            BR      160$                    ;GO TO 160$ IF NO ERROR
  5287  017242  000240                            NOP                     ;RETURN HERE IF ERROR
  5288  017244  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
  5289  017246  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
  5290  017250  000137  017274                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
  5291  017254                                160$:
  5292  017254  004737  040640                      JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
  5293  017260  000405                            BR      170$                    ;GO TO 170$ IF NO ERROR
  5294  017262  000240                            NOP                     ;RETURN HERE IF ERROR
  5295  017264  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
  5296  017266  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
  5297  017270  000137  017274                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
  5298  017274                                170$:
  5299  017274                                180$:
  5300
  5301                                        ;;*****************************************************************
  5302                                        ;*TEST 16       FORMAT W/ MID TRANSFER SEEK
  5303                                        ;;*****************************************************************
  5304  017274                                ST16:
  5305  017274  000004                            SCOPE                           ;SCOPE CALL
  5306  017276  000240                            NOP                             ;START OF TEST
  5307  017300  012706  001100                    MOV     #STACK,SP               ;INITIALIZE STACK POINTER
  5308  017304  013700  001276                    MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
  5309  017310  013701  001446                    MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
  5310  017314  012737  000016  001226            MOV     #16,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
  5311  017322                                10$:
  5312
  5313                                        ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
  5314  017322  012737  000000  001432            MOV     #0,RMDC0                        ;CYLINDER = 0
  5315  017330  012737  002037  001404            MOV     #002037,RMDA0                   ;START AT LAST TRACK & SECTOR
  5316  017336  012737  010000  001430            MOV     #FMT16,RMOF0                    ;16 BIT FORMAT
  5317  017344  012737  176774  001400            MOV     #<^C<<2+256.>*2>+1>,RMWC0;WORD COUNT FOR 2 SECTORS
  5318  017352  012737  102574  001402            MOV     #BUFONE,RMBA0                   ;DATA BUFFER ADDRESS
  5319  017360  012737  000062  001376            MOV     #WH,RMCS1O                      ;WRITE HEADER AND DATA
  5320  017366  012737  064110  001174            MOV     #ZEROS,$TMP0                    ;USE ALL ZEROS DATA PATTERN
  5321  017374  012737  000001  001176            MOV     #1,$TMP1
```

```
5322  017402  004737  036114            JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
5323  017406                          20$:
5324
5325                                  ;PREPARE DEVICE FOR DATA TRANSFER
5326  017406  004737  033274            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5327  017412  154130                    .WORD   154130  ;TASK DESCRIPTOR
5328  017414  000404                    BR      30$                     ;GO TO 30$ IF NO ERROR
5329  017416  000240                    NOP                     ;RETURN HERE IF ERROR
5330  017420  104000                    ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5331  017422  000137  020050            JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5332  017426                          30$:
5333
5334                                  ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE ON CYLINDER
5335  017426  012737  000005  001376      MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
5336  017434  012702  001533            MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
5337  017440  112722  000006            MOVB    #RMDA,(R2)+
5338  017444  112722  000034            MOVB    #RMDC,(R2)+
5339  017450  112722  000032            MOVB    #RMOF,(R2)+
5340  017454  112722  000000            MOVB    #RMCS1,(R2)+
5341  017460  112722  000200            MOVB    #200,(R2)+
5342  017464  004737  037262            JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5343  017470  000404                    BR      40$     ;GO TO 40$ IF NO ERROR
5344  017472  000240                    NOP                     ;RETURN HERE IF ERROR
5345  017474  104000                    ERROR                   ;ERROR DEFINED BY PUT SUB
5346  017476  000137  020050            JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5347  017502                          40$:
5348
5349                                  ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
5350  017502  004737  036726            JSR     PC,GETSTS               ;SETUP FOR STATUS
5351  017506  004737  037622            JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
5352  017512                          50$:
5353
5354                                  ;GO READ SEEK STATUS
5355  017512  004737  037012            JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5356  017516  000404                    BR      60$     ;GO TO 60$ IF NO ERROR
5357  017520  000240                    NOP                     ;RETURN HERE IF ERROR
5358  017522  104000                    ERROR                   ;ERROR DEFINED BY GET SUB
5359  017524  000137  020050            JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5360  017530                          60$:
5361
5362                                  ;VERIFY THE RESULTS OF THE SEEK COMMAND
5363  017530  004737  044714            JSR     PC,SEKSTS       ;GO VERIFY RESULTS OF SEEK OPERATION
5364  017534  000405                    BR      70$                     ;GO TO 70$ IF NO ERROR
5365  017536  000240                    NOP                     ;RETURN HERE IF ERROR
5366  017540  104000                    ERROR                   ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5367  017542  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5368  017544  000137  020050            JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5369  017550                          70$:
5370
5371                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5372  017550  012737  000063  001376      MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
5373  017556  012702  001536            MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
5374  017562  112722  000002            MOVB    #RMWC,(R2)+
5375  017566  112722  000004            MOVB    #RMBA,(R2)+
5376  017572  112722  000000            MOVB    #RMCS1,(R2)+
5377  017576  112722  000200            MOVB    #200,(R2)+
```

```
5378  017602  004737  037262                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5379  017606  000404                          BR      80$     ;GO TO 80$ IF NO ERROR
5380  017610  000240                          NOP                     ;RETURN HERE IF ERROR
5381  017612  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
5382  017614  000137  020050                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5383  017620                          80$:
5384
5385
5386  017620  004737  037622          ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
                                              JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
5387  017624  004737  037012                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5388  017630  000404                          BR      90$     ;GO TO 90$ IF NO ERROR
5389  017632  000240                          NOP                     ;RETURN HERE IF ERROR
5390  017634  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5391  017636  000137  020050                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5392  017642                          90$:
5393
5394
5395  017642  004737  040006          ;VERIFY RESULTS OF WRITE COMMAND
                                              JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5396  017646  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
5397  017650  000240                          NOP                     ;RETURN HERE IF ERROR
5398  017652  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5399  017654  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5400  017656  000137  020050                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5401  017662                          100$:
5402  017662  004737  052312                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5403  017666  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
5404  017670  000240                          NOP                     ;RETURN HERE IF ERROR
5405  017672  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5406  017674  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5407  017676  000137  020050                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5408  017702                          110$:
5409  017702  004737  040640                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5410  017706  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
5411  017710  000240                          NOP                     ;RETURN HERE IF ERROR
5412  017712  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5413  017714  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5414  017716  000137  020050                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5415  017722                          120$:
5416
5417
5418                                  ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
5419  017722  012737  000053  001376          MOV     #WCH!GO,RMCS1O          ;READ HEADER & DATA COMMAND
5420  017730  004737  037262                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5421  017734  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
5422  017736  000240                          NOP                     ;RETURN HERE IF ERROR
5423  017740  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
5424  017742  000137  020050                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5425  017746                          130$:
5426
5427                                  ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
5428  017746  004737  037622                  JSR     PC,TIMOUT               ;WAIT FOR WRITE CHECK TO FINISH
5429  017752  004737  037012                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5430  017756  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
5431  017760  000240                          NOP                     ;RETURN HERE IF ERROR
5432  017762  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
5433  017764  000137  020050                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
```

```
5434  017770                          140$:
5435
5436                                  ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
5437  017770  004737  040006                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5438  017774  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
5439  017776  000240                        NOP                     ;RETURN HERE IF ERROR
5440  020000  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5441  020002  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5442  020004  000137  020050                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5443  020010                          150$:
5444  020010  004737  052312                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5445  020014  000405                        BR      160$                    ;GO TO 160$ IF NO ERROR
5446  020016  000240                        NOP                     ;RETURN HERE IF ERROR
5447  020020  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5448  020022  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5449  020024  000137  020050                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5450  020030                          160$:
5451  020030  004737  040640                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5452  020034  000405                        BR      170$                    ;GO TO 170$ IF NO ERROR
5453  020036  000240                        NOP                     ;RETURN HERE IF ERROR
5454  020040  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5455  020042  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5456  020044  000137  020050                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5457  020050                          170$:
5458  020050                          180$:
5459
5460                                  ;;*****************************************************************
5461                                  ;*TEST 17       FORMAT W/ IMPLIED SEEK
5462                                  ;;*****************************************************************
5463  020050                          TST17:
5464  020050  000004                        SCOPE                   ;SCOPE CALL
5465  020052  000240                        NOP                     ;START OF TEST
5466  020054  012706  001100                MOV     #STACK,SP       ;INITIALIZE STACK POINTER
5467  020060  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
5468  020064  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
5469  020070  012737  000017  001226        MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5470  020076                          10$:
5471
5472                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5473  020076  012737  000000  001432        MOV     #0,RMDC0                ;CYLINDER = 0
5474  020104  012737  000000  001404        MOV     #0,RMDA0                ;TRACK = SECTOR = 0
5475  020112  012737  010000  001430        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
5476  020120  012737  176774  001400        MOV     #<^C<<2+256.>*2>+1>,RMWC0;WORD COUNT FOR 2 SECTORS
5477  020126  012737  102574  001402        MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
5478  020134  012737  000062  001376        MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
5479  020142  012737  064110  001174        MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
5480  020150  012737  000001  001176        MOV     #1,$TMP1
5481  020156  004737  036114                JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
5482  020162                            $:
5483
5484                                  ;PREPARE DEVICE FOR DATA TRANSFER
5485  020162  004737  033274                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
5486  020166  154130                        .WORD   154130  ;TASK DESCRIPTOR
5487  020170  000404                        BR      30$                     ;GO TO 30$ IF NO ERROR
5488  020172  000240                        NOP                     ;RETURN HERE IF ERROR
5489  020174  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
```

```
5490  020176  000137  020646                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5491  020202                                30$:
5492
5493                                        ;SETUP PARAMETERS AND EXECUTE SEEK TO GET DRIVE OFF CYLINDER
5494  020202  013737  001432  020650              MOV     RMDC0,190$              ;SAVE CYLINDER ADDRESS
5495  020210  012737  001466  001432              MOV     #822.,RMDC0             ;SEEK TO LAST CYLINDER
5496  020216  012737  000005  001376              MOV     #SEEK!GO,RMCS10         ;CHANGE COMMAND TO SEEK
5497  020224  012702  001533                      MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
5498  020230  112722  000006                      MOVB    #RMDA,(R2)+
5499  020234  112722  000034                      MOVB    #RMDC,(R2)+
5500  020240  112722  000032                      MOVB    #RMOF,(R2)+
5501  020244  112722  000000                      MOVB    #RMCS1,(R2)+
5502  020250  112722  000200                      MOVB    #200,(R2)+
5503  020254  004737  037262                      JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5504  020260  000404                              BR      40$     ;GO TO 40$ IF NO ERROR
5505  020262  000240                              NOP                             ;RETURN HERE IF ERROR
5506  020264  104000                              ERROR                           ;ERROR DEFINED BY PUT SUB
5507  020266  000137  020646                      JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5508  020272                                40$:
5509
5510                                        ;SETUP FOR READING STATUS AND THEN WAIT FOR SEEK TO COMPLETE
5511  020272  004737  036726                      JSR     PC,GETSTS               ;SETUP FOR STATUS
5512  020276  004737  037622                      JSR     PC,TIMOUT               ;WAIT FOR SEEK TO COMPLETE
5513  020302                                50$:
5514
5515                                        ;GO READ SEEK STATUS
5516  020302  004737  037012                      JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
5517  020306  000404                              BR      60$     ;GO TO 60$ IF NO ERROR
5518  020310  000240                              NOP                             ;RETURN HERE IF ERROR
5519  020312  104000                              ERROR                           ;ERROR DEFINED BY GET SUB
5520  020314  000137  020646                      JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5521  020320                                60$:
5522
5523                                        ;VERIFY THE RESULTS OF THE SEEK COMMAND
5524  020320  004737  044714                      JSR     PC,SEKSTS               ;GO VERIFY RESULTS OF SEEK OPERATION
5525  020324  000405                              BR      70$                             ;GO TO 70$ IF NO ERROR
5526  020326  000240                              NOP                             ;RETURN HERE IF ERROR
5527  020330  104000                              ERROR                           ;ERROR # DEFINED BY SEKSTS SUBROUTINE
5528  020332  004736                              JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5529  020334  000137  020646                      JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5530  020340                                70$:
5531
5532                                        ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
5533  020340  013737  020650  001432              MOV     190$,RMDC0              ;RESTORE DISK ADDRESS
5534  020346  012737  000063  001376              MOV     #WH!GO,RMCS10           ;LOAD WRITE HEADER AND DATA
5535  020354  012702  001536                      MOV     #PUTINX+3,R2            ;EXTEND REGISTER INDEX TABLE
5536  020360  112722  000002                      MOVB    #RMWC,(R2)+
5537  020364  112722  000004                      MOVB    #RMBA,(R2)+
5538  020370  112722  000000                      MOVB    #RMCS1,(R2)+
5539  020374  112722  000200                      MOVB    #200,(R2)+
5540  020400  004737  037262                      JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
5541  020404  000404                              BR      80$     ;GO TO 80$ IF NO ERROR
5542  020406  000240                              NOP                             ;RETURN HERE IF ERROR
5543  020410  104000                              ERROR                           ;ERROR DEFINED BY PUT SUB
5544  020412  000137  020646                      JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5545  020416                                80$:
```

```
5546
5547                                    ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
5548  020416  004737  037622                    JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
5549  020422  004737  037012                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5550  020426  000404                            BR      90$     ;GO TO 90$ IF NO ERROR
5551  020430  000240                            NOP                             ;RETURN HERE IF ERROR
5552  020432  104000                            ERROR                           ;ERROR DEFINED BY GET SUB
5553  020434  000137  020646                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5554  020440                                90$:
5555
5556                                    ;VERIFY RESULTS OF WRITE COMMAND
5557  020440  004737  040006                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5558  020444  000405                            BR      100$                    ;GO TO 100$ IF NO ERROR
5559  020446  000240                            NOP                             ;RETURN HERE IF ERROR
5560  020450  104000                            ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
5561  020452  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5562  020454  000137  020646                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5563  020460                                100$:
5564  020460  004737  052312                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5565  020464  000405                            BR      110$                    ;GO TO 110$ IF NO ERROR
5566  020466  000240                            NOP                             ;RETURN HERE IF ERROR
5567  020470  104000                            ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
5568  020472  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5569  020474  000137  020646                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5570  020500                                110$:
5571  020500  004737  040640                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5572  020504  000405                            BR      120$                    ;GO TO 120$ IF NO ERROR
5573  020506  000240                            NOP                             ;RETURN HERE IF ERROR
5574  020510  104000                            ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
5575  020512  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5576  020514  000137  020646                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
5577  020520                                120$:
5578
5579
5580                                    ;WRITE CHECK HEADER AND DATA FOR SECTORS JUST WRITTEN
5581  020520  012737  000053  001376            MOV     #WCH!GO,RMCS1           ;READ HEADER & DATA COMMAND
5582  020526  004737  037262                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5583  020532  000404                            BR      130$    ;GO TO 130$ IF NO ERROR
5584  020534  000240                            NOP                             ;RETURN HERE IF ERROR
5585  020536  104000                            ERROR                           ;ERROR DEFINED BY PUT SUB
5586  020540  000137  020646                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5587  020544                                130$:
5588
5589                                    ;WAIT FOR WRITE CHECK TO COMPLETE AND GET STATUS
5590  020544  004737  037622                    JSR     PC,TIMOUT               ;WAIT FOR WRITE CHECK TO FINISH
5591  020550  004737  037012                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5592  020554  000404                            BR      140$    ;GO TO 140$ IF NO ERROR
5593  020556  000240                            NOP                             ;RETURN HERE IF ERROR
5594  020560  104000                            ERROR                           ;ERROR DEFINED BY GET SUB
5595  020562  000137  020646                    JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
5596  020566                                140$:
5597
5598                                    ;VERIFY THE RESULTS OF WRITE CHECK OPERATION
5599  020566  004737  040006                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5600  020572  000405                            BR      150$                    ;GO TO 150$ IF NO ERROR
5601  020574  000240                            NOP                             ;RETURN HERE IF ERROR
```

```
5602  020576  104000                          ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
5603  020600  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5604  020602  000137  020646                  JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
5605  020606                          150$:
5606  020606  004737  052312                  JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
5607  020612  000405                          BR      160$                            ;GO TO 160$ IF NO ERROR
5608  020614  000240                          NOP                             ;RETURN HERE IF ERROR
5609  020616  104000                          ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
5610  020620  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5611  020622  000137  020646                  JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
5612  020626                          160$:
5613  020626  004737  040640                  JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
5614  020632  000405                          BR      170$                            ;GO TO 170$ IF NO ERROR
5615  020634  000240                          NOP                             ;RETURN HERE IF ERROR
5616  020636  104000                          ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
5617  020640  004736                          JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
5618  020642  000137  020646                  JMP     180$                            ;GO TO 180$ IF ERROR WAS FOUND
5619  020646                          170$:
5620  020646                          180$:
5621  020646  000401                          BR      200$
5622  020650  000000                  190$:   .WORD                                   ;TEMPORARY STORAGE
5623  020652                          200$:
5624
5625                                  ;;*****************************************************************
5626                                  ;;TEST 20       FORMAT EACH SECTOR ADDRESS
5627                                  ;;*****************************************************************
5628  020652                          TST20:
5629  020652  000004                          SCOPE                           ;SCOPE CALL
5630  020654  000240                          NOP                             ;START OF TEST
5631  020656  012706  001100                  MOV     #STACK,SP               ;INITIALIZE STACK POINTER
5632  020662  013700  001276                  MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
5633  020666  013701  001446                  MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
5634  020672  012737  000020  001226          MOV     #20,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
5635  020700                          10$:
5636
5637                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
5638  020700  012737  000000  001432          MOV     #0,RMDC0                        ;CYLINDER = 0
5639  020706  012737  000000  001404          MOV     #0,RMDA0                        ;TRACK = SECTOR = 0
5640  020714  012737  010000  001430  15$:    MOV     #FMT16,RMOF0                    ;16 BIT FORMAT
5641  020722  012737  177376  001400          MOV     #<^C<2+256.>+1>,RMWC0           ;2 + 256 WORDS
5642  020730  012737  102574  001402          MOV     #BUFONE,RMBA0                   ;DATA BUFFER ADDRESS
5643  020736  012737  000062  001376          MOV     #WH,RMCS10                      ;WRITE HEADER AND DATA
5644  020744  012737  001404  001174          MOV     #RMDA0,$TMP0                    ;USE SECTOR FOR DATA PATTERN
5645  020752  012737  000001  001176          MOV     #1,$TMP1
5646  020760  004737  036114                  JSR     PC,GENBUF                       ;GO GENERATE DATA BUFFER
5647  020764                          20$:
5648
5649                                  ;PREPARE DEVICE FOR DATA TRANSFER
5650  020764  004737  033274                  JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
5651  020770  154130                          .WORD   154130  ;TASK DESCRIPTOR
5652  020772  000404                          BR      30$                             ;GO TO 30$ IF NO ERROR
5653  020774  000240                          NOP                             ;RETURN HERE IF ERROR
5654  020776  104000                          ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
5655  021000  000137  021376                  JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
5656  021004                          30$:
5657
```

```
                                        ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
021004  012737  000063  001376            MOV     #WH!GO,RMCS1O           ;WRITE HEADER AND DATA
021012  012702  001533                    MOV     #PUT,MX,R2              ;WRITE REGISTER INDEX TABLE
021016  112722  000006                    MOVB    #RMDA,(R2)+
021022  112722  000034                    MOVB    #RMDC,(R2)+
021026  112722  000032                    MOVB    #RMOF,(R2)+
021032  112722  000002                    MOVB    #RMWC,(R2)+
021036  112722  000004                    MOVB    #RMBA,(R2)+
021042  112722  000000                    MOVB    #RMCS1,(R2)+
021046  112722  000200                    MOVB    #200,(R2)+
021052  004737  037262                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
021056  000404                            BR      80$     ;GO TO 80$ IF NO ERROR
021060  000240                            NOP                     ;RETURN HERE IF ERROR
021062  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
021064  000137  021376                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
021070                                80$:

                                        ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
021070  004737  036726                    JSR     PC,GETSTS

                                        ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
021074  004737  037622                    JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
021100  004737  037012                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
021104  000404                            BR      90$     ;GO TO 90$ IF NO ERROR
021106  000240                            NOP                     ;RETURN HERE IF ERROR
021110  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
021112  000137  021376                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
021116                                90$:

                                        ;VERIFY RESULTS OF WRITE COMMAND
021116  004737  040006                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
021122  000405                            BR      100$                    ;GO TO 100$ IF NO ERROR
021124  000240                            NOP                     ;RETURN HERE IF ERROR
021126  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
021130  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
021132  000137  021376                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
021136                                100$:
021136  004737  052312                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
021142  000405                            BR      110$                    ;GO TO 110$ IF NO ERROR
021144  000240                            NOP                     ;RETURN HERE IF ERROR
021146  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
021150  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
021152  000137  021376                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
021156                                110$:
021156  004737  040640                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
021162  000405                            BR      120$                    ;GO TO 120$ IF NO ERROR
021164  000240                            NOP                     ;RETURN HERE IF ERROR
021166  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
021170  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
021172  000137  021376                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
021176                                120$:


                                        ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
021176  012737  000073  001376            MOV     #RH!GO,RMCS1O           ;READ HEADER & DATA COMMAND
021204  012737  103600  001402            MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
```

```
5714  021212  004737  037262          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5715  021216  000404                  BR      130$    ;GO TO 130$ IF NO ERROR
5716  021220  000240                  NOP             ;RETURN HERE IF ERROR
5717  021222  104000                  ERROR           ;ERROR DEFINED BY PUT SUB
5718  021224  000137  021376          JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5719  021230                  130$:
5720
5721                          ;WAIT FOR READ TO COMPLETE AND GET STATUS
5722  021230  004737  037622          JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
5723  021234  004737  037012          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5724  021240  000404                  BR      140$    ;GO TO 140$ IF NO ERROR
5725  021242  000240                  NOP             ;RETURN HERE IF ERROR
5726  021244  104000                  ERROR           ;ERROR DEFINED BY GET SUB
5727  021246  000137  021376          JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5728  021252                  140$:
5729
5730                          ;VERIFY THE RESULTS OF READ OPERATION
5731  021252  004737  040006          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5732  021256  000405                  BR      150$            ;GO TO 150$ IF NO ERROR
5733  021260  000240                  NOP             ;RETURN HERE IF ERROR
5734  021262  104000                  ERROR           ;ERROR # DEFINED BY PRIERR SUBROUTINE
5735  021264  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5736  021266  000137  021376          JMP     190$            ;GO TO 190$ IF ERROR WAS FOUND
5737  021272                  150$:
5738  021272  004737  052312          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5739  021276  000405                  BR      160$            ;GO TO 160$ IF NO ERROR
5740  021300  000240                  NOP             ;RETURN HERE IF ERROR
5741  021302  104000                  ERROR           ;ERROR # DEFINED BY DTASTS SUBROUTINE
5742  021304  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5743  021306  000137  021376          JMP     190$            ;GO TO 190$ IF ERROR WAS FOUND
5744  021312                  160$:
5745  021312  004737  040640          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5746  021316  000405                  BR      170$            ;GO TO 170$ IF NO ERROR
5747  021320  000240                  NOP             ;RETURN HERE IF ERROR
5748  021322  104000                  ERROR           ;ERROR # DEFINED BY SECERR SUBROUTINE
5749  021324  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5750  021326  000137  021376          JMP     190$            ;GO TO 190$ IF ERROR WAS FOUND
5751  021332                  170$:
5752
5753                          ;VERIFY DATA
5754  021332  004737  036360          JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
5755  021336  102574                  .WORD   BUFONE          ;STARTING ADDRESS OF WRITE BUFFER
5756  021340  103600                  .WORD   BUFTWO          ;STARTING ADDDRESS OF READ BUFFER
5757  021342  000404                  BR      180$            ;GO TO 180$ IF NO ERROR
5758  021344  000240                  NOP             ;RETURN HERE IF ERROR
5759  021346  104000                  ERROR           ;ERROR # DEFINED BY CMPBUF SUBROUTINE
5760  021350  000137  021376          JMP     190$            ;GO TO 190$ IF ERROR WAS FOUND
5761  021354                  180$:
5762
5763                          ;INCREMENT ADDRESS AND FORMAT NEXT SECTOR
5764  021354  062737  000001  001404          ADD     #1,RMDA0                ;ADVANCE SECTOR COUNT
5765  021362  122737  000037  001404          CMPB    #31.,RMDA0              ;DONE ALL SECTORS??
5766  021370  103402                          BLO     190$                    ;YES!!
5767  021372  000137  020714                  JMP     15$                     ;GO DO NEXT SECTOR
5768  021376                  190$:
5769
```

```
                        ;;*****************************************************************
                        ;*TEST 21       FORMAT EACH TRACK ADDRESS
                        ;;*****************************************************************
021376                  TST21:
021376  000004                SCOPE                   ;SCOPE CALL
021400  000240                NOP                     ;START OF TEST
021402  012706  001100        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
021406  013700  001276        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
021412  013701  001446        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
021416  012737  000021  001226        MOV     #21,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
021424                  10$:

                        ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
021424  012737  000000  001432        MOV     #0,RMDC0                ;CYLINDER = 0
021432  012737  000000  001404        MOV     #0,RMDA0                ;TRACK = SECTOR = 0
021440  012737  010000  001430  15$:   MOV     #FMT16,RMOF0            ;16 BIT FORMAT
021446  012737  177376  001400        MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
021454  012737  102574  001402        MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
021462  012737  000062  001376        MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
021470  012737  001404  001174        MOV     #RMDA0,$TMP0            ;USE TRACK FOR DATA PATTERN
021476  012737  000001  001176        MOV     #1,$TMP1
021504  004737  036114                JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
021510                  20$:

                        ;PREPARE DEVICE FOR DATA TRANSFER
021510  004737  033274        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
021514  154130                .WORD   154130  ;TASK DESCRIPTOR
021516  000404                BR      30$             ;GO TO 30$ IF NO ERROR
021520  000240                NOP                     ;RETURN HERE IF ERROR
021522  104000                ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
021524  000137  022122        JMP     190$            ;GO TO 190$ IF ERROR WAS FOUND
021530                  30$:

                        ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
021530  012737  000063  001376        MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
021536  012702  001533                MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
021542  112722  000006                MOVB    #RMDA,(R2)+
021546  112722  000034                MOVB    #RMDC,(R2)+
021552  112722  000032                MOVB    #RMOF,(R2)+
021556  112722  000002                MOVB    #RMWC,(R2)+
021562  112722  000004                MOVB    #RMBA,(R2)+
021566  112722  000000                MOVB    #RMCS1,(R2)+
021572  112722  000200                MOVB    #200,(R2)+
021576  004737  037262                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
021602  000404                BR      80$     ;GO TO 80$ IF NO ERROR
021604  000240                NOP                     ;RETURN HERE IF ERROR
021606  104000                ERROR                   ;ERROR DEFINED BY PUT SUB
021610  000137  022122        JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
021614                  80$:

                        ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
021614  004737  036726        JSR     PC,GETSTS

                        ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
021620  004737  037622        JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
021624  004737  037012        JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
```

```
5826  021630  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
5827  021632  000240                        NOP                     ;RETURN HERE IF ERROR
5828  021634  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
5829  021636  000137  022122                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5830  021642                        90$:
5831
5832                                  ;VERIFY RESULTS OF WRITE COMMAND
5833  021642  004737  040006                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5834  021646  000405                        BR      100$                    ;GO TO 100$ IF NO ERROR
5835  021650  000240                        NOP                     ;RETURN HERE IF ERROR
5836  021652  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5837  021654  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5838  021656  000137  022122                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5839  021662                        100$:
5840  021662  004737  052312                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
5841  021666  000405                        BR      110$                    ;GO TO 110$ IF NO ERROR
5842  021670  000240                        NOP                     ;RETURN HERE IF ERROR
5843  021672  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
5844  021674  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5845  021676  000137  022122                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5846  021702                        110$:
5847  021702  004737  040640                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
5848  021706  000405                        BR      120$                    ;GO TO 120$ IF NO ERROR
5849  021710  000240                        NOP                     ;RETURN HERE IF ERROR
5850  021712  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
5851  021714  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5852  021716  000137  022122                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
5853  021722                        120$:
5854
5855
5856                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
5857  021722  012737  000073  001376          MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
5858  021730  012737  103600  001402          MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
5859  021736  004737  037262                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
5860  021742  000404                        BR      130$    ;GO TO 130$ IF NO ERROR
5861  021744  000240                        NOP                     ;RETURN HERE IF ERROR
5862  021746  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
5863  021750  000137  022122                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5864  021754                        130$:
5865
5866                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
5867  021754  004737  037622                JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
5868  021760  004737  037012                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
5869  021764  000404                        BR      140$    ;GO TO 140$ IF NO ERROR
5870  021766  000240                        NOP                     ;RETURN HERE IF ERROR
5871  021770  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
5872  021772  000137  022122                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
5873  021776                        140$:
5874
5875                                  ;VERIFY THE RESULTS OF READ OPERATION
5876  021776  004737  040006                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
5877  022002  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
5878  022004  000240                        NOP                     ;RETURN HERE IF ERROR
5879  022006  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
5880  022010  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
5881  022012  000137  022122                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
```

```
 5882  022016                          150$:
 5883  022016  004737  052312                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 5884  022022  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
 5885  022024  000240                          NOP                     ;RETURN HERE IF ERROR
 5886  022026  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 5887  022030  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 5888  022032  000137  022122                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 5889  022036                          160$:
 5890  022036  004737  040640                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 5891  022042  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
 5892  022044  000240                          NOP                     ;RETURN HERE IF ERROR
 5893  022046  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 5894  022050  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 5895  022052  000137  022122                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 5896  022056                          170$:
 5897
 5898                                  ;VERIFY DATA
 5899  022056  004737  036360                  JSR     PC,CMPBUF       ;GO COMPARE WRITE, READ DATA BUFFERS
 5900  022062  102574                          .WORD   BUFONE          ;STARTING ADDRESS OF WRITE BUFFER
 5901  022064  103600                          .WORD   BUFTWO          ;STARTING ADDDRESS OF READ BUFFER
 5902  022066  000404                          BR      180$                    ;GO TO 180$ IF NO ERROR
 5903  022070  000240                          NOP                     ;RETURN HERE IF ERROR
 5904  022072  104000                          ERROR                   ;ERROR # DEFINED BY CMPBUF SUBROUTINE
 5905  022074  000137  022122                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 5906  022100                          180$:
 5907
 5908                                  ;INCREMENT ADDRESS AND FORMAT NEXT TRACK
 5909  022100  062737  000400  001404          ADD     #400,RMDAO              ;ADVANCE TRACK COUNT
 5910  022106  022737  002000  001404          CMP     #2000,RMDAO             ;DONE ALL TRACKS??
 5911  022114  103402                          BLO     190$                    ;YES!!
 5912  022116  000137  021440                  JMP     15$                     ;GO DO NEXT SECTOR
 5913  022122                          190$:
 5914
 5915                                  ;;*****************************************************************
 5916                                  ;*TEST 22       FORMAT PRIME CYLINDERS
 5917                                  ;;*****************************************************************
 5918  022122                          TST22:
 5919  022122  000004                          SCOPE                   ;SCOPE CALL
 5920  022124  000240                          NOP                     ;START OF TEST
 5921  022126  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
 5922  022132  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 5923  022136  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 5924  022142  012737  000022  001226          MOV     #22,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 5925  022150                          10$:
 5926
 5927                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
 5928  022150  012737  000001  001432          MOV     #1,RMDC0                ;CYLINDER = 0
 5929  022156  012737  000000  001404          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
 5930  022164  012737  010000  001430  15$:    MOV     #FMT16,RMOF0            ;16 BIT FORMAT
 5931  022172  012737  177376  001400          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
 5932  022200  012737  102574  001402          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
 5933  022206  012737  000062  001376          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
 5934  022214  012737  001432  001174          MOV     #RMDC0,$TMP0            ;USE CYLINDER FOR DATA PATTERN
 5935  022222  012737  000001  001176          MOV     #1,$TMP1
 5936  022230  004737  036114                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
 5937  022234                          20$:
```

```
 5938
 5939                                   ;PREPARE DEVICE FOR DATA TRANSFER
 5940  022234  004737  033274                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 5941  022240  154130                            .WORD   154130  ;TASK DESCRIPTOR
 5942  022242  000404                            BR      30$                     ;GO TO 30$ IF NO ERROR
 5943  022244  000240                            NOP                     ;RETURN HERE IF ERROR
 5944  022246  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 5945  022250  000137  022644                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 5946  022254                                 30$:
 5947
 5948                                   ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
 5949  022254  012737  000063  001376            MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
 5950  022262  012702  001533                    MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
 5951  022266  112722  000006                    MOVB    #RMDA,(R2)+
 5952  022272  112722  000034                    MOVB    #RMDC,(R2)+
 5953  022276  112722  000032                    MOVB    #RMOF,(R2)+
 5954  022302  112722  000002                    MOVB    #RMWC,(R2)+
 5955  022306  112722  000004                    MOVB    #RMBA,(R2)+
 5956  022312  112722  000000                    MOVB    #RMCS1,(R2)+
 5957  022316  112722  000200                    MOVB    #200,(R2)+
 5958  022322  004737  037262                    JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
 5959  022326  000404                            BR      80$     ;GO TO 80$ IF NO ERROR
 5960  022330  000240                            NOP                     ;RETURN HERE IF ERROR
 5961  022332  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
 5962  022334  000137  022644                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
 5963  022340                                 80$:
 5964
 5965                                   ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
 5966  022340  004737  036726                    JSR     PC,GETSTS
 5967
 5968                                   ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
 5969  022344  004737  037622                    JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
 5970  022350  004737  037012                    JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
 5971  022354  000404                            BR      90$     ;GO TO 90$ IF NO ERROR
 5972  022356  000240                            NOP                     ;RETURN HERE IF ERROR
 5973  022360  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
 5974  022362  000137  022644                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
 5975  022366                                 90$:
 5976
 5977                                   ;VERIFY RESULTS OF WRITE COMMAND
 5978  022366  004737  040006                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 5979  022372  000405                            BR      100$                    ;GO TO 100$ IF NO ERROR
 5980  022374  000240                            NOP                     ;RETURN HERE IF ERROR
 5981  022376  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 5982  022400  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 5983  022402  000137  022644                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 5984  022406                                 100$:
 5985  022406  004737  052312                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 5986  022412  000405                            BR      110$                    ;GO TO 110$ IF NO ERROR
 5987  022414  000240                            NOP                     ;RETURN HERE IF ERROR
 5988  022416  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 5989  022420  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 5990  022422  000137  022644                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 5991  022426                                 110$:
 5992  022426  004737  040640                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 5993  022432  000405                            BR      120$                    ;GO TO 120$ IF NO ERROR
```

```
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
                                120$:

                                ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
                                        MOV     #RH!GO,RMCS10                   ;READ HEADER & DATA COMMAND
                                        MOV     #BUFTWO,RMBA0                   ;CHANGE BUS ADDRESS
                                        JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
                                        BR      130$    ;GO TO 130$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR DEFINED BY PUT SUB
                                        JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
                                130$:

                                ;WAIT FOR READ TO COMPLETE AND GET STATUS
                                        JSR     PC,TIMOUT                       ;WAIT FOR READ TO COMPLETE
                                        JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
                                        BR      140$    ;GO TO 140$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR DEFINED BY GET SUB
                                        JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
                                140$:

                                ;VERIFY THE RESULTS OF READ OPERATION
                                        JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
                                        BR      150$                            ;GO TO 150$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
                                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
                                150$:
                                        JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
                                        BR      160$                            ;GO TO 160$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
                                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
                                160$:
                                        JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
                                        BR      170$                            ;GO TO 170$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
                                170$:

                                ;VERIFY DATA
                                        JSR     PC,CMPBUF               ;GO COMPARE WRITE, READ DATA BUFFERS
                                        .WORD   BUFONE                  ;STARTING ADDRESS OF WRITE BUFFER
                                        .WORD   BUFTWO                  ;STARTING ADDDRESS OF READ BUFFER
                                        BR      180$                            ;GO TO 180$ IF NO ERROR
                                        NOP                             ;RETURN HERE IF ERROR
                                        ERROR                           ;ERROR # DEFINED BY CMPBUF SUBROUTINE
```

```
 6050  022620  000137  022644            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 6051  022624                          180$:
 6052
 6053                                  ;INCREMENT ADDRESS AND FORMAT NEXT PRIME CYLINDER
 6054  022624  006337  001432            ASL     RMDC0                   ;ADVANCE CYLINDER COUNT
 6055  022630  022737  001466  001432      CMP     #822.,RMDC0             ;DONE ALL CYLINDERS??
 6056  022636  103402                      BLO     190$                    ;YES!!
 6057  022640  000137  022164            JMP     15$                     ;GO DO NEXT CYLINDER
 6058  022644                          190$:
 6059
 6060                                  ;;*****************************************************************
 6061                                  ;*TEST 23        FORMAT LAST SECTOR
 6062                                  ;;*****************************************************************
 6063  022644                          TST23:
 6064  022644  000004                     SCOPE                           ;SCOPE CALL
 6065  022646  000240                     NOP                             ;START OF TEST
 6066  022650  012706  001100             MOV     #STACK,SP               ;INITIALIZE STACK POINTER
 6067  022654  013700  001276             MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
 6068  022660  013701  001446             MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
 6069  022664  012737  000023  001226     MOV     #23,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
 6070  022672                          10$:
 6071
 6072                                  ;PREPARE DEVICE FOR DATA TRANSFER
 6073  022672  004737  033274             JSR     PC,TSTPRP               ;PREPARE DEVICE FOR TEST
 6074  022676  154130                      .WORD   154130  ;TASK DESCRIPTOR
 6075  022700  000404                      BR      30$                     ;GO TO 30$ IF NO ERROR
 6076  022702  000240                      NOP                             ;RETURN HERE IF ERROR
 6077  022704  104000                      ERROR                           ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 6078  022706  000137  023142             JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 6079  022712                          30$:
 6080
 6081                                  ;SETUP PARAMETERS FOR READING LAST SECTOR
 6082  022712  012737  001466  001432      MOV     #822.,RMDC0             ;LAST CYLINDER
 6083  022720  012737  002037  001404      MOV     #2037,RMDA0             ;TRACK = 4 SECTOR = 31.
 6084  022726  012737  010000  001430      MOV     #FMT16,RMOF0            ;16 BIT FORMAT
 6085  022734  012737  177376  001400      MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
 6086  022742  012737  103600  001402      MOV     #BUFTW0,RMBA0           ;DATA BUFFER ADDRESS
 6087  022750  012737  000073  001376      MOV     #RH!GO,RMCS10           ;WRITE HEADER AND DATA
 6088
 6089  022756  012702  001533              MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
 6090  022762  112722  000006              MOVB    #RMDA,(R2)+
 6091  022766  112722  000034              MOVB    #RMDC,(R2)+
 6092  022772  112722  000032              MOVB    #RMOF,(R2)+
 6093  022776  112722  000004              MOVB    #RMBA,(R2)+
 6094  023002  112722  000002              MOVB    #RMWC,(R2)+
 6095  023006  112722  000000              MOVB    #RMCS1,(R2)+
 6096  023012  112722  000200              MOVB    #200,(R2)+
 6097
 6098
 6099  023016                          120$:
 6100
 6101                                  ;READ HEADER AND DATA OF LAST SECTOR
 6102  023016  004737  037262             JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
 6103  023022  000404                      BR      130$    ;GO TO 130$ IF NO ERROR
 6104  023024  000240                      NOP                             ;RETURN HERE IF ERROR
 6105  023026  104000                      ERROR                           ;ERROR DEFINED BY PUT SUB
```

```
                        JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
130$:
                ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
                        JSR     PC,GETSTS
                ;WAIT FOR READ TO COMPLETE AND GET STATUS
                        JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
                        JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
                        BR      140$    ;GO TO 140$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR DEFINED BY GET SUB
                        JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
140$:
                ;VERIFY THE RESULTS OF READ OPERATION
                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
                        BR      150$                    ;GO TO 150$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
150$:
                        JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
                        BR      160$                    ;GO TO 160$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
160$:
                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
                        BR      170$                    ;GO TO 170$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
170$:

190$:

                ;;*****************************************************************
                ;*TEST 24        FORMAT W/ AOE ERROR
                ;;*****************************************************************
TST24:
                        SCOPE                   ;SCOPE CALL
                        NOP                     ;START OF TEST
                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                        MOV     #24,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

                ;PREPARE DEVICE FOR DATA TRANSFER
                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                        .WORD   154130  ;TASK DESCRIPTOR
                        BR      30$                     ;GO TO 30$ IF NO ERROR
                        NOP                     ;RETURN HERE IF ERROR
                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
```

```
                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
30$:

                    ;SETUP PARAMETERS FOR READING 2 SECTORS STARTING WITH LAST SECTOR
                        MOV     #822.,RMDC0             ;START AT LAST CYLINDER
                        MOV     #2037,RMDA0             ;TRACK = 4 SECTOR = 31.
                        MOV     #FMT16,RMOF0            ;16 BIT FORMAT
                        MOV     #<^C<<2+256.>*2>+1>,RMWC0       ;FORMAT 2 SECTORS
                        MOV     #BUFTW0,RMBA0           ;DATA BUFFER ADDRESS
                        MOV     #RH!GO,RMCS10           ;WRITE HEADER AND DATA
                        MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
                        MOVB    #RMDA,(R2)+
                        MOVB    #RMDC,(R2)+
                        MOVB    #RMOF,(R2)+
                        MOVB    #RMWC,(R2)+
                        MOVB    #RMBA,(R2)+
                        MOVB    #RMCS1,(R2)+
                        MOVB    #200,(R2)+
80$:

125$:
                        JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
                        BR      130$    ;GO TO 130$ IF NO ERROR
                        NOP                             ;RETURN HERE IF ERROR
                        ERROR                           ;ERROR DEFINED BY PUT SUB
                        JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
130$:

                    ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
                        JSR     PC,GETSTS

                    ;WAIT FOR READ TO COMPLETE AND GET STATUS
                        JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
                        JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
                        BR      140$    ;GO TO 140$ IF NO ERROR
                        NOP                             ;RETURN HERE IF ERROR
                        ERROR                           ;ERROR DEFINED BY GET SUB
                        JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
140$:

                    ;VERIFY THE RESULTS OF READ OPERATION
                        JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
                        BR      150$                            ;GO TO 150$ IF NO ERROR
                        NOP                             ;RETURN HERE IF ERROR
                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                        JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
150$:
                        JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
                        BR      160$                            ;GO TO 160$ IF NO ERROR
                        NOP                             ;RETURN HERE IF ERROR
                        ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
                        JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
                        JMP     190$                            ;GO TO 190$ IF ERROR WAS FOUND
160$:
                        JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
```

```
                                        BR      170$                    ;GO TO 170$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
                                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
                                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
                                170$:

                                190$:

                                ;;*****************************************************************
                                ;*TEST 25       FORMAT INVALID SECTOR ADDRESS
                                ;;*****************************************************************
                                TST25:
                                        SCOPE                   ;SCOPE CALL
                                        NOP                     ;START OF TEST
                                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
                                        MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
                                        MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
                                        MOV     #25,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
                                10$:

                                ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
                                        MOV     #0,RMDC0                ;CYLINDER = 0
                                        MOV     #30.,RMDA0              ;TRACK = 0, SECTOR = 30
                                15$:    MOV     #0,RMOF0                ;18 BIT FORMAT
                                        MOV     #^C<2+256.>+1>,RMWC0    ;2 + 256 WORDS
                                        MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
                                        MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
                                        MOV     #RMDA0,$TMP0            ;USE SECTOR FOR DATA PATTERN
                                        MOV     #1,$TMP1
                                        JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
                                20$:

                                ;PREPARE DEVICE FOR DATA TRANSFER
                                        JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
                                        .WORD   154130  ;TASK DESCRIPTOR
                                        BR      30$                     ;GO TO 30$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
                                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
                                30$:

                                ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
                                        MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
                                        MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
                                        MOVB    #RMDA,(R2)+
                                        MOVB    #RMDC,(R2)+
                                        MOVB    #RMOF,(R2)+
                                        MOVB    #RMWC,(R2)+
                                        MOVB    #RMBA,(R2)+
                                        MOVB    #RMCS1,(R2)+
                                        MOVB    #200,(R2)+
                                        JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
                                        BR      80$     ;GO TO 80$ IF NO ERROR
                                        NOP                     ;RETURN HERE IF ERROR
                                        ERROR                   ;ERROR DEFINED BY PUT SUB
```

```
T25     FORMAT INVALID SECTOR ADDRESS

6274  023652  000137  024166                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6275  023656                              80$:
6276
6277                                      ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6278  023656  004737  036726                JSR     PC,GETSTS
6279
6280                                      ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6281  023662  004737  037622                JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
6282  023666  004737  037012                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6283  023672  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
6284  023674  000240                        NOP                     ;RETURN HERE IF ERROR
6285  023676  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
6286  023700  000137  024166                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6287  023704                              90$:
6288
6289                                      ;VERIFY RESULTS OF WRITE COMMAND
6290  023704  004737  040006                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6291  023710  000405                        BR      100$                    ;GO TO 100$ IF NO ERROR
6292  023712  000240                        NOP                     ;RETURN HERE IF ERROR
6293  023714  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6294  023716  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6295  023720  000137  024166                JMP     190$                    ;GO TO 130$ IF ERROR WAS FOUND
6296  023724                              100$:
6297  023724  004737  052312                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6298  023730  000405                        BR      110$                    ;GO TO 110$ IF NO ERROR
6299  023732  000240                        NOP                     ;RETURN HERE IF ERROR
6300  023734  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6301  023736  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6302  023740  000137  024166                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6303  023744                              110$:
6304  023744  004737  040640                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6305  023750  000405                        BR      120$                    ;GO TO 120$ IF NO ERROR
6306  023752  000240                        NOP                     ;RETURN HERE IF ERROR
6307  023754  104000                        ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6308  023756  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6309  023760  000137  024166                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6310  023764                              120$:
6311
6312                                      ;CLEAR "IAE" ERROR AND PREPARE DEVICE
6313  023764  004737  033274                JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6314  023770  154130                        .WORD   154130  ;TASK DESCRIPTOR
6315  023772  000404                        BR      125$                    ;GO TO 125$ IF NO ERROR
6316  023774  000240                        NOP                     ;RETURN HERE IF ERROR
6317  023776  104000                        ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6318  024000  000137  024166                JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6319  024004                              125$:
6320
6321                                      ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
6322  024004  012737  000073  001376        MOV     #RH!GO,RMCS10                   ;READ HEADER & DATA COMMAND
6323  024012  012737  103600  001402        MOV     #BUFTWO,RMBA0                   ;CHANGE BUS ADDRESS
6324  024020  004737  037262                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6325  024024  000404                        BR      130$    ;GO TO 130$ IF NO ERROR
6326  024026  000240                        NOP                     ;RETURN HERE IF ERROR
6327  024030  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
6328  024032  000137  024166                JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6329  024036                              130$:
```

```
6330
6331                                    ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6332  024036  004737  036726                    JSR     PC,GETSTS
6333
6334                                    ;WAIT FOR READ TO COMPLETE AND GET STATUS
6335  024042  004737  037622                    JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
6336  024046  004737  037012                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6337  024052  000404                            BR      140$    ;GO TO 140$ IF NO ERROR
6338  024054  000240                            NOP                             ;RETURN HERE IF ERROR
6339  024056  104000                            ERROR                           ;ERROR DEFINED BY GET SUB
6340  024060  000137  024166                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6341  024064                                140$:
6342
6343                                    ;VERIFY THE RESULTS OF READ OPERATION
6344  024064  004737  040006                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6345  024070  000405                            BR      150$                    ;GO TO 150$ IF NO ERROR
6346  024072  000240                            NOP                             ;RETURN HERE IF ERROR
6347  024074  104000                            ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
6348  024076  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6349  024100  000137  024166                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6350  024104                                150$:
6351  024104  004737  052312                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6352  024110  000405                            BR      160$                    ;GO TO 160$ IF NO ERROR
6353  024112  000240                            NOP                             ;RETURN HERE IF ERROR
6354  024114  104000                            ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
6355  024116  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6356  024120  000137  024166                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6357  024124                                160$:
6358  024124  004737  040640                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6359  024130  000405                            BR      170$                    ;GO TO 170$ IF NO ERROR
6360  024132  000240                            NOP                             ;RETURN HERE IF ERROR
6361  024134  104000                            ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
6362  024136  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6363  024140  000137  024166                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6364  024144                                170$:
6365  024144                                180$:
6366
6367                                    ;INCREMENT ADDRESS AND FORMAT NEXT SECTOR
6368  024144  062737  000001  001404            ADD     #1,RMDA0                ;ADVANCE SECTOR COUNT
6369  024152  022737  000037  001404            CMP     #31.,RMDA0              ;DONE ALL SECTORS??
6370  024160  103402                            BLO     190$                    ;YES!!
6371  024162  000137  023502                    JMP     15$                     ;GO DO NEXT SECTOR
6372  024166                                190$:
6373
6374                                    ;;*****************************************************************
6375                                    ;*TEST 26       FORMAT INVALID TRACK ADDRESS
6376                                    ;;*****************************************************************
6377  024166                            T$T26:
6378  024166  000004                            SCOPE                           ;SCOPE CALL
6379  024170  000240                            NOP                             ;START OF TEST
6380  024172  012706  001100                    MOV     #STACK,SP               ;INITIALIZE STACK POINTER
6381  024176  013700  001276                    MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
6382  024202  013701  001446                    MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
6383  024206  012737  000026  001226            MOV     #26,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
6384  024214                                10$:
6385
```

```
                                ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
024214  012737  000000  001432            MOV     #0,RMDC0                ;CYLINDER = 0
024222  012737  002400  001404            MOV     #2400,RMDA0             ;TRACK = 5,SECTOR = 0
024230  012737  010000  001430  15$:      MOV     #FMT16,RMOF0            ;16 BIT FORMAT
024236  012737  177376  001400            MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
024244  012737  102574  001402            MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
024252  012737  000062  001376            MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
024260  012737  001404  001174            MOV     #RMDA0,$TMP0            ;USE SECTOR FOR DATA PATTERN
024266  012737  000001  001176            MOV     #1,$TMP1
024274  004737  036114                    JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
024300                          20$:

                                ;PREPARE DEVICE FOR DATA TRANSFER
024300  004737  033274                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
024304  154130                            .WORD   154130  ;TASK DESCRIPTOR
024306  000404                            BR      30$             ;GO TO 30$ IF NO ERROR
024310  000240                            NOP                     ;RETURN HERE IF ERROR
024312  104000                            ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
024314  000137  024714                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
024320                          30$:

                                ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
024320  012737  000063  001376            MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
024326  012702  001533                    MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
024332  112722  000006                    MOVB    #RMDA,(R2)+
024336  112722  000034                    MOVB    #RMDC,(R2)+
024342  112722  000032                    MOVB    #RMOF,(R2)+
024346  112722  000002                    MOVB    #RMWC,(R2)+
024352  112722  000004                    MOVB    #RMBA,(R2)+
024356  112722  000000                    MOVB    #RMCS1,(R2)+
024362  112722  000200                    MOVB    #200,(R2)+
024366  004737  037262                    JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
024372  000404                            BR      80$     ;GO TO 80$ IF NO ERROR
024374  000240                            NOP                     ;RETURN HERE IF ERROR
024376  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
024400  000137  024714                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
024404                          80$:

                                ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
024404  004737  036726                    JSR     PC,GETSTS

                                ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
024410  004737  037622                    JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
024414  004737  037012                    JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
024420  000404                            BR      90$     ;GO TO 90$ IF NO ERROR
024422  000240                            NOP                     ;RETURN HERE IF ERROR
024424  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
024426  000137  024714                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
024432                          90$:

                                ;VERIFY RESULTS OF WRITE COMMAND
024432  004737  040706                    JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
024436  000405                            BR      100$                    ;GO TO 100$ IF NO ERROR
024440  000240                            NOP                     ;RETURN HERE IF ERROR
024442  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
024444  004736                            JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
```

```
T26     FORMAT INVALID TRACK ADDRESS

  6442  024446  000137  024714            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
  6443  024452                    100$:
  6444  024452  004737  052312            JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
  6445  024456  000405                    BR      110$                    ;GO TO 110$ IF NO ERROR
  6446  024460  000240                    NOP                     ;RETURN HERE IF ERROR
  6447  024462  104000                    ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
  6448  024464  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
  6449  024466  000137  024714            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
  6450  024472                    110$:
  6451  024472  004737  040640            JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
  6452  024476  000405                    BR      120$                    ;GO TO 120$ IF NO ERROR
  6453  024500  000240                    NOP                     ;RETURN HERE IF ERROR
  6454  024502  104000                    ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
  6455  024504  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
  6456  024506  000137  024714            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
  6457  024512                    120$:
  6458
  6459                            ;CLEAR "IAE" ERROR AND PREPARE DEVICE
  6460  024512  004737  033274            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
  6461  024516  154130                    .WORD   154130  ;TASK DESCRIPTOR
  6462  024520  000404                    BR      125$                    ;GO TO 125$ IF NO ERROR
  6463  024522  000240                    NOP                     ;RETURN HERE IF ERROR
  6464  024524  104000                    ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
  6465  024526  000137  024714            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
  6466  024532                    125$:
  6467
  6468                            ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
  6469  024532  012737  000073  001376            MOV     #RH!GO,RMCS1            ;READ HEADER & DATA COMMAND
  6470  024540  012737  103600  001402            MOV     #BUFTWO,RMBA            ;CHANGE BUS ADDRESS
  6471  024546  004737  037262                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
  6472  024552  000404                    BR      130$    ;GO TO 130$ IF NO ERROR
  6473  024554  000240                    NOP                     ;RETURN HERE IF ERROR
  6474  024556  104000                    ERROR                   ;ERROR DEFINED BY PUT SUB
  6475  024560  000137  024714            JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
  6476  024564                    130$:
  6477
  6478                            ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
  6479  024564  004737  036726            JSR     PC,GETSTS
  6480
  6481                            ;WAIT FOR READ TO COMPLETE AND GET STATUS
  6482  024570  004737  037622            JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
  6483  024574  004737  037012            JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
  6484  024600  000404                    BR      140$    ;GO TO 140$ IF NO ERROR
  6485  024602  000240                    NOP                     ;RETURN HERE IF ERROR
  6486  024604  104000                    ERROR                   ;ERROR DEFINED BY GET SUB
  6487  024606  000137  024714            JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
  6488  024612                    140$:
  6489
  6490                            ;VERIFY THE RESULTS OF READ OPERATION
  6491  024612  004737  040006            JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
  6492  024616  000405                    BR      150$                    ;GO TO 150$ IF NO ERROR
  6493  024620  000240                    NOP                     ;RETURN HERE IF ERROR
  6494  024622  104000                    ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
  6495  024624  004736                    JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
  6496  024626  000137  024714            JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
  6497  024632                    150$:
```

```
6498  024632  004737  052312          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6499  024636  000405                  BR      160$                    ;GO TO 160$ IF NO ERROR
6500  024640  000240                  NOP                     ;RETURN HERE IF ERROR
6501  024642  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6502  024644  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6503  024646  000137  024714          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6504  024652                  160$:
6505  024652  004737  040640          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6506  024656  000405                  BR      170$                    ;GO TO 170$ IF NO ERROR
6507  024660  000240                  NOP                     ;RETURN HERE IF ERROR
6508  024662  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6509  024664  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6510  024666  000137  024714          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6511  024672                  170$:
6512  024672                  180$:
6513
6514                          ;INCREMENT ADDRESS AND FORMAT NEXT TRACK
6515  024672  062737  000400  001404          ADD     #400,RMDAO              ;ADVANCE TRACK COUNT
6516  024700  022737  003400  001404          CMP     #3400,RMDAO             ;DONE ALL TRACKS??
6517  024706  103402                  BLO     190$                    ;YES!!
6518  024710  000137  024230          JMP     15$                     ;GO DO NEXT TRACK
6519  024714                  190$:
6520
6521                          ;;*****************************************************************
6522                          ;*TEST 27       FORMAT INVALID CYLINDER ADDRESS
6523                          ;;*****************************************************************
6524  024714                  TST27:
6525  024714  000004                  SCOPE                   ;SCOPE CALL
6526  024716  000240                  NOP                     ;START OF TEST
6527  024720  012706  001100          MOV     #STACK,SP       ;INITIALIZE STACK POINTER
6528  024724  013700  001276          MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
6529  024730  013701  001446          MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
6530  024734  012737  000027  001226          MOV     #27,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6531  024742                  10$:
6532
6533                          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
6534  024742  012737  001467  001432          MOV     #823.,RMDC0             ;START AT LAST CYLINDER + 1
6535  024750  012737  000000  001404          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
6536  024756  012737  010000  001430  15$:     MOV     #FMT16,RMOF0            ;16 BIT FORMAT
6537  024764  012737  177376  001400          MOV     #<^C<2^256.>+1>,RMWC0   ;2 + 256 WORDS
6538  024772  012737  102574  001402          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
6539  025000  012737  000062  001376          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
6540  025006  012737  001432  001174          MOV     #RMDC0,$TMP0            ;USE CYLINDER FOR DATA PATTERN
6541  025014  012737  000001  001176          MOV     #1,$TMP1
6542  025022  004737  036114          JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
6543  025026                  20$:
6544
6545                          ;PREPARE DEVICE FOR DATA TRANSFER
6546  025026  004737  033274          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6547  025032  154130                  .WORD   154130  ;TASK DESCRIPTOR
6548  025034  000404                  BR      30$                     ;GO TO 30$ IF NO ERROR
6549  025036  000240                  NOP                     ;RETURN HERE IF ERROR
6550  025040  104000                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6551  025042  000137  025442          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6552  025046                  30$:
6553
```

```
                                ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
025046  012737  000063  001376            MOV     #WH!GO,RMCS1O           ;WRITE HEADER AND DATA
025054  012702  001533                    MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
025060  112722  000006                    MOVB    #RMDA,(R2)+
025064  112722  000034                    MOVB    #RMDC,(R2)+
025070  112722  000032                    MOVB    #RMOF,(R2)+
025074  112722  000002                    MOVB    #RMWC,(R2)+
025100  112722  000004                    MOVB    #RMBA,(R2)+
025104  112722  000000                    MOVB    #RMCS1,(R2)+
025110  112722  000200                    MOVB    #200,(R2)+
025114  004737  037262                    JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
025120  000404                            BR      80$     ;GO TO 80$ IF NO ERROR
025122  000240                            NOP                     ;RETURN HERE IF ERROR
025124  104000                            ERROR                   ;ERROR DEFINED BY PUT SUB
025126  000137  025442                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
025132                          80$:

                                ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
025132  004737  036726                    JSR     PC,GETSTS

                                ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
025136  004737  037622                    JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
025142  004737  037012                    JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
025146  000404                            BR      90$     ;GO TO 90$ IF NO ERROR
025150  000240                            NOP                     ;RETURN HERE IF ERROR
025152  104000                            ERROR                   ;ERROR DEFINED BY GET SUB
025154  000137  025442                    JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
025160                          90$:

                                ;VERIFY RESULTS OF WRITE COMMAND
025160  004737  040006                    JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
025164  000405                            BR      100$                    ;GO TO 100$ IF NO ERROR
025166  000240                            NOP                     ;RETURN HERE IF ERROR
025170  104000                            ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
025172  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
025174  000137  025442                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
025200                          100$:
025200  004737  052312                    JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
025204  000405                            BR      110$                    ;GO TO 110$ IF NO ERROR
025206  000240                            NOP                     ;RETURN HERE IF ERROR
025210  104000                            ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
025212  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
025214  000137  025442                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
025220                          110$:
025220  004737  040640                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
025224  000405                            BR      120$                    ;GO TO 120$ IF NO ERROR
025226  000240                            NOP                     ;RETURN HERE IF ERROR
025230  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
025232  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
025234  000137  025442                    JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
025240                          120$:

                                ;CLEAR "IAE" ERROR AND PREPARE DEVICE
025240  004737  033274                    JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
025244  154130                            .WORD   154130  ;TASK DESCRIPTOR
025246  000404                            BR      125$                    ;GO TO 125$ IF NO ERROR
```

```
 6610  025250  000240                          NOP                     ;RETURN HERE IF ERROR
 6611  025252  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 6612  025254  000137  025442                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 6613  025260                          125$:
 6614
 6615                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
 6616  025260  012737  000073  001376          MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
 6617  025266  012737  103600  001402          MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
 6618  025274  004737  037262                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 6619  025300  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
 6620  025302  000240                          NOP                     ;RETURN HERE IF ERROR
 6621  025304  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
 6622  025306  000137  025442                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
 6623  025312                          130$:
 6624
 6625                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
 6626  025312  004737  036726                  JSR     PC,GETSTS
 6627
 6628                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
 6629  025316  004737  037622                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
 6630  025322  004737  037012                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 6631  025326  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
 6632  025330  000240                          NOP                     ;RETURN HERE IF ERROR
 6633  025332  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
 6634  025334  000137  025442                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
 6635  025340                          140$:
 6636
 6637                                  ;VERIFY THE RESULTS OF READ OPERATION
 6638  025340  004737  040006                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 6639  025344  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
 6640  025346  000240                          NOP                     ;RETURN HERE IF ERROR
 6641  025350  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 6642  025352  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 6643  025354  000137  025442                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 6644  025360                          150$:
 6645  025360  004737  052312                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 6646  025364  000405                          BR      160$                    ;GO TO 160$ IF NO ERROR
 6647  025366  000240                          NOP                     ;RETURN HERE IF ERROR
 6648  025370  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 6649  025372  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 6650  025374  000137  025442                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 6651  025400                          160$:
 6652  025400  004737  040640                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 6653  025404  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
 6654  025406  000240                          NOP                     ;RETURN HERE IF ERROR
 6655  025410  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 6656  025412  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 6657  025414  000137  025442                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 6658  025420                          170$:
 6659
 6660                                  ;INCREMENT ADDRESS AND FORMAT NEXT CYLINDER
 6661  025420  062737  000001  001432          ADD     #1,RMDCO                ;ADVANCE CYLINDER COUNT
 6662  025426  022737  001777  001432          CMP     #1777,RMDCO             ;DONE ALL CYLINDERS??
 6663  025434  103402                          BLO     190$                    ;YES!!
 6664  025436  000137  024756                  JMP     15$                     ;GO DO NEXT SECTOR
 6665  025442                          190$:
```

```
                                ;;*****************************************************************
                                ;*TEST 30        FORMAT AT OFFSET
                                ;;*****************************************************************
025442                          TST30:
025442  000004                          SCOPE                   ;SCOPE CALL
025444  000240                          NOP                     ;START OF TEST
025446  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
025452  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
025456  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
025462  012737  000030  001226          MOV     #30,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
025470                          10$:

                                ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
025470  012737  000000  001432          MOV     #0,RMDC0                ;CYLINDER = 0
025476  012737  000000  001404          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
025504  012737  010000  001430          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
025512  012737  177376  001400          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
025520  012737  102574  001402          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
025526  012737  000063  001376          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
025534  012737  001404  001174          MOV     #RMDA0,$TMP0            ;USE SECTOR FOR DATA PATTERN
025542  012737  000001  001176          MOV     #1,$TMP1
025550  004737  036114          15$:    JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
025554                          20$:

                                ;PREPARE DEVICE FOR DATA TRANSFER
025554  004737  033274                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
025560  154130                          .WORD   154130  ;TASK DESCRIPTOR
025562  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
025564  000240                          NOP                     ;RETURN HERE IF ERROR
025566  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
025570  000137  026334                  JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
025574                          30$:

                                ;OFFSET DEVICE FOR WRITE
025574  012737  000015  001376          MOV     #OFFSET!GO,RMCS10       ;CHANGE TO OFFSET COMMAND
025602  012702  001533                  MOV     #PUTINX,R2              ;WRITE PUT INDEX TABLE
025606  112722  000006                  MOVB    #RMDA,(R2)+
025612  112722  000034                  MOVB    #RMDC,(R2)+
025616  112722  000032                  MOVB    #RMOF,(R2)+
025622  112722  000000                  MOVB    #RMCS1,(R2)+
025626  112712  000200                  MOVB    #200,(R2)               ;TERMINATE TABLE
025632  004737  037262                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
025636  000404                          BR      35$     ;GO TO 35$ IF NO ERROR
025640  000240                          NOP                     ;RETURN HERE IF ERROR
025642  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
025644  000137  026334                  JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
025650                          35$:

                                ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND AT OFFSET
025650  012737  000063  001376          MOV     #WH!GO,RMCS10           ; WRITE HEADER AND DATA COMMAND
025656  012702  001533                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
025662  112722  000002                  MOVB    #RMWC,(R2)+
025666  112722  000004                  MOVB    #RMBA,(R2)+
025672  112722  000000                  MOVB    #RMCS1,(R2)+
025676  112722  000200                  MOVB    #200,(R2)+
```

```
6722  025702  004737  037262          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6723  025706  000404                  BR      80$     ;GO TO 80$ IF NO ERROR
6724  025710  000240                  NOP                     ;RETURN HERE IF ERROR
6725  025712  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
6726  025714  000137  026334          JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6727  025720                  80$:
6728
6729                          ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6730  025720  004737  036726          JSR     PC,GETSTS
6731
6732                          ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6733  025724  004737  037622          JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
6734  025730  004737  037012          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6735  025734  000404                  BR      90$     ;GO TO 90$ IF NO ERROR
6736  025736  000240                  NOP                     ;RETURN HERE IF ERROR
6737  025740  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
6738  025742  000137  026334          JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6739  025746                  90$:
6740
6741                          ;VERIFY RESULTS OF WRITE COMMAND
6742  025746  004737  040006          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6743  025752  000405                  BR      100$                    ;GO TO 100$ IF NO ERROR
6744  025754  000240                  NOP                     ;RETURN HERE IF ERROR
6745  025756  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6746  025760  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6747  025762  000137  026334          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6748  025766                  100$:
6749  025766  004737  052312          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
6750  025772  000405                  BR      110$                    ;GO TO 110$ IF NO ERROR
6751  025774  000240                  NOP                     ;RETURN HERE IF ERROR
6752  025776  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
6753  026000  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6754  026002  000137  026334          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6755  026006                  110$:
6756  026006  004737  040640          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6757  026012  000405                  BR      120$                    ;GO TO 120$ IF NO ERROR
6758  026014  000240                  NOP                     ;RETURN HERE IF ERROR
6759  026016  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6760  026020  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6761  026022  000137  026334          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6762  026026                  120$:
6763                          ;OFFSET DEVICE FOR READ
6764  026026  012737  000015  001376          MOV     #OFFSET!GO,RMCS1        ;CHANGE TO OFFSET COMMAND
6765  026034  012702  001533          MOV     #PUTINX,R2              ;WRITE PUT INDEX TABLE
6766  026040  112722  000006          MOVB    #RMDA,(R2)+
6767  026044  112722  000034          MOVB    #RMDC,(R2)+
6768  026050  112722  000032          MOVB    #RMOF,(R2)+
6769  026054  112722  000000          MOVB    #RMCS1,(R2)+
6770  026060  112722  000200          MOVB    #200,(R2)+
6771  026064  004737  037262          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6772  026070  000404                  BR      125$    ;GO TO 125$ IF NO ERROR
6773  026072  000240                  NOP                     ;RETURN HERE IF ERROR
6774  026074  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
6775  026076  000137  026334          JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6776  026102                  125$:
6777
```

```
                                        ;READ HEADER AND DATA AT OFFSET
026102  012737  000073  001376                MOV     #RH!GO,RMCS1O           ;READ HEADER & DATA COMMAND
026110  012737  103600  001402                MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
026116  012702  001533                        MOV     #PUTINX,R2              ;WRITE PUT INDEX TABLE
026122  112722  000002                        MOVB    #RMWC,(R2)+
026126  112722  000004                        MOVB    #RMBA,(R2)+
026132  112722  000000                        MOVB    #RMCS1,(R2)+
026136  112722  000200                        MOVB    #200,(R2)+
026142  004737  037262                        JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
026146  000404                                BR      130$    ;GO TO 130$ IF NO ERROR
026150  000240                                NOP                             ;RETURN HERE IF ERROR
026152  104000                                ERROR                           ;ERROR DEFINED BY PUT SUB
026154  000137  026334                        JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
026160                                  130$:

                                        ;WAIT FOR READ TO COMPLETE AND GET STATUS
026160  004737  037622                        JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
026164  004737  037012                        JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
026170  000404                                BR      140$    ;GO TO 140$ IF NO ERROR
026172  000240                                NOP                             ;RETURN HERE IF ERROR
026174  104000                                ERROR                           ;ERROR DEFINED BY GET SUB
026176  000137  026334                        JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
026202                                  140$:

                                        ;VERIFY THE RESULTS OF READ OPERATION
026202  004737  040006                        JSR     PC,PRIERR               ;GO CHECK FOR PRIMARY ERRORS
026206  000405                                BR      150$                    ;GO TO 150$ IF NO ERROR
026210  000240                                NOP                             ;RETURN HERE IF ERROR
026212  104000                                ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
026214  004736                                JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
026216  000137  026334                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
026222                                  150$:
026222  004737  052312                        JSR     PC,DTASTS               ;GO VERIFY RESULTS OF DATA TRANSFER
026226  000405                                BR      160$                    ;GO TO 160$ IF NO ERROR
026230  000240                                NOP                             ;RETURN HERE IF ERROR
026232  104000                                ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
026234  004736                                JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
026236  000137  026334                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
026242                                  160$:
026242  004737  040640                        JSR     PC,SECERR               ;GO CHECK FOR SECONDARY ERRORS
026246  000405                                BR      170$                    ;GO TO 170$ IF NO ERROR
026250  000240                                NOP                             ;RETURN HERE IF ERROR
026252  104000                                ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
026254  004736                                JSR     PC,@(SP)+               ;GO BACK FOR MORE ERROR CHECKS
026256  000137  026334                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
026262                                  170$:

                                        ;VERIFY DATA
026262  004737  036360                        JSR     PC,CMPBUF               ;GO COMPARE WRITE, READ DATA BUFFERS
026266  102574                                .WORD   BUFONE                  ;STARTING ADDRESS OF WRITE BUFFER
026270  103600                                .WORD   BUFTWO                  ;STARTING ADDDRESS OF READ BUFFER
026272  000404                                BR      180$                    ;GO TO 180$ IF NO ERROR
026274  000240                                NOP                             ;RETURN HERE IF ERROR
026276  104000                                ERROR                           ;ERROR # DEFINED BY CMPBUF SUBROUTINE
026300  000137  026334                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
```

```
6834  026304
6835  026304  032737  000200  001430          180$:   BIT     #OFD,RMOFO              ;DONE BOTH OFFSETS??
6836  026312  001010                                  BNE     190$                    ;YES!!
6837  026314  052737  000200  001430                  BIS     #OFD,RMOFO              ;SET FORWARD OFFSET
6838  026322  012737  102574  001402                  MOV     #BUFONE,RMBAO           ;CHANGE DATA BUFFER
6839  026330  000137  025554                          JMP     20$
6840  026334                                  190$:
6841
6842                                          ;;*****************************************************************
6843                                          ;*TEST 31        IVC FORMAT TEST
6844                                          ;;*****************************************************************
6845  026334                                  TST31:
6846  026334  000004                                  SCOPE                           ;SCOPE CALL
6847  026336  000240                                  NOP                             ;START OF TEST
6848  026340  012706  001100                          MOV     #STACK,SP               ;INITIALIZE STACK POINTER
6849  026344  013700  001276                          MOV     $BASE,R0                ;R0=UNIBUS ADDRESS
6850  026350  013701  001446                          MOV     TSTQUE,R1               ;(R1) = DEVICE BEING TESTED
6851  026354  012737  000031  001226                  MOV     #31,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
6852  026362                                  10$:
6853
6854                                          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
6855  026362  012737  000000  001432                  MOV     #0,RMDC0                ;CYLINDER = 0
6856  026370  012737  000000  001404                  MOV     #0,RMDA0                ;TRACK = SECTOR = 0
6857  026376  012737  010000  001430                  MOV     #FMT16,RMOFO            ;16 BIT FORMAT
6858  026404  012737  177376  001400                  MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
6859  026412  012737  102574  001402                  MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
6860  026420  012737  000062  001376                  MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
6861  026426  012737  001404  001174                  MOV     #RMDA0,$TMP0            ;USE SECTOR FOR DATA PATTERN
6862  026434  012737  000001  001176                  MOV     #1,$TMP1
6863  026442  004737  036114                  15$:    JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
6864  026446                                  20$:
6865
6866                                          ;PREPARE DEVICE FOR DATA TRANSFER
6867  026446  004737  033274                          JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6868  026452  154130                                  .WORD   154130  ;TASK DESCRIPTOR
6869  026454  000404                                  BR      30$                     ;GO TO 30$ IF NO ERROR
6870  026456  000240                                  NOP                     ;RETURN HERE IF ERROR
6871  026460  104000                                  ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
6872  026462  000137  027142                          JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6873  026466                                  30$:
6874
6875                                          ;RESET VOLUME VALID BY SETTING AND RESETTING DIAGNOSTIC MODE
6876  026466  012737  000001  001422                  MOV     #DMD,RMMR10             ;SET DIAGNOSTIC MODE
6877  026474  112737  000024  001533                  MOVB    #RMMR1,PUTINX           ;WRITE REGISTER INDEX TABLE
6878  026502  112737  000200  001534                  MOVB    #200,PUTINX+1
6879  026510  004737  037262                          JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
6880  026514  000404                                  BR      35$     ;GO TO 35$ IF NO ERROR
6881  026516  000240                                  NOP                             ;RETURN HERE IF ERROR
6882  026520  104000                                  ERROR                           ;ERROR DEFINED BY PUT SUB
6883  026522  000137  027142                          JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6884  026526                                  35$:
6885
6886                                          ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
6887  026526  012737  000063  001376                  MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
6888  026534  012737  000000  001422                  MOV     #0,RMMR10               ;RESET DIAGNOSTIC MODE
6889  026542  012702  001533                          MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
```

```
6890  026546  112722  000024              MOVB    #RMMR1,(R2)+
6891  026552  112722  000006              MOVB    #RMDA,(R2)+
6892  026556  112722  000034              MOVB    #RMDC,(R2)+
6893  026562  112722  000032              MOVB    #RMOF,(R2)+
6894  026566  112722  000002              MOVB    #RMWC,(R2)+
6895  026572  112722  000004              MOVB    #RMBA,(R2)+
6896  026576  112722  000000              MOVB    #RMCS1,(R2)+
6897  026602  112722  000200              MOVB    #200,(R2)+
6898  026606  004737  037262              JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
6899  026612  000404                      BR      80$     ;GO TO 80$ IF NO ERROR
6900  026614  000240                      NOP                     ;RETURN HERE IF ERROR
6901  026616  104000                      ERROR                   ;ERROR DEFINED BY PUT SUB
6902  026620  000137  027142              JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6903  026624                      80$:
6904
6905                              ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
6906  026624  004737  036726              JSR     PC,GETSTS
6907
6908                              ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
6909  026630  004737  037622              JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
6910  026634  004737  037012              JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
6911  026640  000404                      BR      90$     ;GO TO 90$ IF NO ERROR
6912  026642  000240                      NOP                     ;RETURN HERE IF ERROR
6913  026644  104000                      ERROR                   ;ERROR DEFINED BY GET SUB
6914  026646  000137  027142              JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
6915  026652                      90$:
6916
6917                              ;VERIFY RESULTS OF WRITE COMMAND
6918  026652  004737  040006              JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
6919  026656  000405                      BR      100$                    ;GO TO 100$ IF NO ERROR
6920  026660  000240                      NOP                     ;RETURN HERE IF ERROR
6921  026662  104000                      ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
6922  026664  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6923  026666  000137  027142              JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6924  026672                      100$:
6925  026672  032737  010000  001370      BIT     #IVC,RMER2I             ;IS IVC STATUS SET??
6926  026700  001012                      BNE     110$                    ;YES!!
6927  026702  013737  001370  001142      MOV     RMER2I,$BDDAT           ;RECEIVED STATUS FOR TYPEOUT
6928  026710  013737  001370  001140      MOV     RMER2I,$GDDAT           ;EXPECTED STATUS
6929  026716  052737  010000  001140      BIS     #IVC,$GDDAT
6930  026724  104342                      ERROR   342                     ;IVC NOT SET DURING FORMAT
6931  026726                      110$:
6932  026726  004737  040640              JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
6933  026732  000405                      BR      120$                    ;GO TO 120$ IF NO ERROR
6934  026734  000240                      NOP                     ;RETURN HERE IF ERROR
6935  026736  104000                      ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
6936  026740  004736                      JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
6937  026742  000137  027142              JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
6938  026746                      120$:
6939
6940                              ;CLEAR IVC ERROR
6941  026746  004737  033274              JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
6942  026752  040100                      .WORD   040100  ;TASK DESCRIPTOR
6943  026754  000404                      BR      125$                    ;GO TO 125$ IF NO ERROR
6944  026756  000240                      NOP                     ;RETURN HERE IF ERROR
6945  026760  104000                      ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
```

```
 6946  026762  000137  027142                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 6947  026766                                125$:
 6948
 6949                                        ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
 6950  026766  012737  000073  001376                MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
 6951  026774  012737  103600  001402                MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
 6952  027002  004737  037262                        JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 6953  027006  000404                                BR      130$    ;GO TO 130$ IF NO ERROR
 6954  027010  000240                                NOP                     ;RETURN HERE IF ERROR
 6955  027012  104000                                ERROR                   ;ERROR DEFINED BY PUT SUB
 6956  027014  000137  027142                        JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
 6957  027020                                130$:
 6958
 6959                                        ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
 6960  027020  004737  036726                        JSR     PC,GETSTS
 6961
 6962                                        ;WAIT FOR READ TO COMPLETE AND GET STATUS
 6963  027024  004737  037622                        JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
 6964  027030  004737  037012                        JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 6965  027034  000404                                BR      140$    ;GO TO 140$ IF NO ERROR
 6966  027036  000240                                NOP                     ;RETURN HERE IF ERROR
 6967  027040  104000                                ERROR                   ;ERROR DEFINED BY GET SUB
 6968  027042  000137  027142                        JMP     190$    ;GO TO 190$ IF ERROR WAS FOUND
 6969  027046                                140$:
 6970
 6971                                        ;VERIFY THE RESULTS OF READ OPERATION
 6972  027046  004737  040006                        JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 6973  027052  000405                                BR      150$                    ;GO TO 150$ IF NO ERROR
 6974  027054  000240                                NOP                     ;RETURN HERE IF ERROR
 6975  027056  104000                                ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 6976  027060  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 6977  027062  000137  027142                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 6978  027066                                150$:
 6979  027066  032737  010000  001370                BIT     #IVC,RMER2I             ;IS IVC STATUS SET??
 6980  027074  001012                                BNE     160$                    ;YES!!
 6981  027076  013737  001370  001142                MOV     RMER2I,$BDDAT           ;RECEIVED STATUS FOR TYPEOUT
 6982  027104  013737  001370  001140                MOV     RMER2I,$GDDAT           ;EXPECTED STATUS
 6983  027112  052737  010000  001140                BIS     #IVC,$GDDAT
 6984  027120  104342                                ERROR   342                     ;IVC NOT SET DURING FORMAT
 6985  027122                                160$:
 6986  027122  004737  040640                        JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 6987  027126  000405                                BR      170$                    ;GO TO 170$ IF NO ERROR
 6988  027130  000240                                NOP                     ;RETURN HERE IF ERROR
 6989  027132  104000                                ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 6990  027134  004736                                JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 6991  027136  000137  027142                        JMP     190$                    ;GO TO 190$ IF ERROR WAS FOUND
 6992  027142                                170$:
 6993  027142                                190$:
 6994
 6995                                        ;;*****************************************************************
 6996                                        ;*TEST 32       FORMAT ERROR TEST - 18
 6997                                        ;;*****************************************************************
 6998  027142                                TST32:
 6999  027142  000004                                SCOPE                   ;SCOPE CALL
 7000  027144  000240                                NOP                     ;START OF TEST
 7001  027146  012706  001100                        MOV     #STACK,SP       ;INITIALIZE STACK POINTER
```

```
 7002  027152  013700  001276            MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
 7003  027156  013701  001446            MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
 7004  027162  012737  000032  001226    MOV     #32,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 7005  027170                          10$:
 7006
 7007                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
 7008  027170  012737  000000  001432    MOV     #0,RMDC0                ;CYLINDER = 0
 7009  027176  012737  000000  001404    MOV     #0,RMDA0                ;TRACK = SECTOR = 0
 7010  027204  012737  000000  001430    MOV     #0,RMOF0                ;18 BIT FORMAT
 7011  027212  012737  177376  001400    MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
 7012  027220  012737  102574  001402    MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
 7013  027226  012737  000062  001376    MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
 7014  027234  012737  064110  001174    MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
 7015  027242  012737  000001  001176    MOV     #1,$TMP1
 7016  027250  004737  036114            JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
 7017  027254                          20$:
 7018
 7019                                  ;PREPARE DEVICE FOR DATA TRANSFER
 7020  027254  004737  033274            JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
 7021  027260  154130                    .WORD   154130  ;TASK DESCRIPTOR
 7022  027262  000404                    BR      30$                     ;GO TO 30$ IF NO ERROR
 7023  027264  000240                    NOP                     ;RETURN HERE IF ERROR
 7024  027266  104000                    ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
 7025  027270  000137  027664            JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 7026  027274                          30$:
 7027
 7028                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
 7029  027274  012737  000063  001376    MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
 7030  027302  012702  001533            MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
 7031  027306  112722  000006            MOVB    #RMDA,(R2)+
 7032  027312  112722  000034            MOVB    #RMDC,(R2)+
 7033  027316  112722  000032            MOVB    #RMOF,(R2)+
 7034  027322  112722  000002            MOVB    #RMWC,(R2)+
 7035  027326  112722  000004            MOVB    #RMBA,(R2)+
 7036  027332  112722  000000            MOVB    #RMCS1,(R2)+
 7037  027336  112722  000200            MOVB    #200,(R2)+
 7038  027342  004737  037262            JSR     PC,PUT                  ;GO WRITE REGISTERS VIA PUT SUB
 7039  027346  000404                    BR      80$     ;GO TO 80$ IF NO ERROR
 7040  027350  000240                    NOP                     ;RETURN HERE IF ERROR
 7041  027352  104000                    ERROR                   ;ERROR DEFINED BY PUT SUB
 7042  027354  000137  027664            JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 7043  027360                          80$:
 7044
 7045                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
 7046  027360  004737  036726            JSR     PC,GETSTS
 7047
 7048                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
 7049  027364  004737  037622            JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
 7050  027370  004737  037012            JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
 7051  027374  000404                    BR      90$     ;GO TO 90$ IF NO ERROR
 7052  027376  000240                    NOP                     ;RETURN HERE IF ERROR
 7053  027400  104000                    ERROR                   ;ERROR DEFINED BY GET SUB
 7054  027402  000137  027664            JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 7055  027406                          90$:
 7056
 7057                                  ;VERIFY RESULTS OF WRITE COMMAND
```

```
 7058  027406  004737  040006          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 7059  027412  000405                  BR      100$                    ;GO TO 100$ IF NO ERROR
 7060  027414  000240                  NOP                     ;RETURN HERE IF ERROR
 7061  027416  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 7062  027420  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7063  027422  000137  027664          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 7064  027426                  100$:
 7065  027426  004737  052312          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
 7066  027432  000405                  BR      110$                    ;GO TO 110$ IF NO ERROR
 7067  027434  000240                  NOP                     ;RETURN HERE IF ERROR
 7068  027436  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
 7069  027440  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7070  027442  000137  027664          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 7071  027446                  110$:
 7072  027446  004737  040640          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
 7073  027452  000405                  BR      120$                    ;GO TO 120$ IF NO ERROR
 7074  027454  000240                  NOP                     ;RETURN HERE IF ERROR
 7075  027456  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
 7076  027460  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7077  027462  000137  027664          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 7078  027466                  120$:
 7079
 7080
 7081                          ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN AT 16 BIT FORMAT
 7082  027466  012737  010000  001430          MOV     #FMT16,RMOF0            ;CHANGE TO 16 BIT FORMAT
 7083  027474  012737  000073  001376          MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
 7084  027502  012737  103600  001402          MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
 7085  027510  004737  037262                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
 7086  027514  000404                  BR      130$    ;GO TO 130$ IF NO ERROR
 7087  027516  000240                  NOP                     ;RETURN HERE IF ERROR
 7088  027520  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
 7089  027522  000137  027664          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 7090  027526                  130$:
 7091
 7092                          ;WAIT FOR READ TO COMPLETE AND GET STATUS
 7093  027526  004737  037622          JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
 7094  027532  004737  037012          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
 7095  027536  000404                  BR      140$    ;GO TO 140$ IF NO ERROR
 7096  027540  000240                  NOP                     ;RETURN HERE IF ERROR
 7097  027542  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
 7098  027544  000137  027664          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
 7099  027550                  140$:
 7100
 7101                          ;VERIFY THE RESULTS OF READ OPERATION
 7102  027550  004737  040006          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
 7103  027554  000405                  BR      150$                    ;GO TO 150$ IF NO ERROR
 7104  027556  000240                  NOP                     ;RETURN HERE IF ERROR
 7105  027560  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
 7106  027562  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
 7107  027564  000137  027664          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
 7108  027570                  150$:
 7109
 7110                          ;VERIFY THAT FORMAT ERROR IS SET
 7111  027570  032737  000020  001342          BIT     #FER,RMER1I             ;IS FORMAT ERROR SET??
 7112  027576  001022                  BNE     160$                    ;YES!!
 7113  027600  004737  052312          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
```

```
7114  027604  000405                          BR      155$                    ;GO TO 155$ IF NO ERROR
7115  027606  000240                          NOP                     ;RETURN HERE IF ERROR
7116  027610  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7117  027612  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7118  027614  000137  027664                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7119  027620  013737  001342  001142  155$:   MOV     RMER1I,$BDDAT           ;BAD DATA FOR TYPEOUT
7120  027626  013737  001342  001140          MOV     RMER1I,$GDDAT           ;EXPECTED DATA
7121  027634  052737  000020  001140          BIS     #FER,$GDDAT
7122  027642  104343                          ERROR   343                     ;FORMAT ERROR NOT SET
7123  027644                          160$:
7124  027644  004737  040640                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7125  027650  000405                          BR      170$                    ;GO TO 170$ IF NO ERROR
7126  027652  000240                          NOP                     ;RETURN HERE IF ERROR
7127  027654  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7128  027656  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7129  027660  000137  027664                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7130  027664                          170$:
7131  027664                          180$:
7132
7133                                  ;;*****************************************************************
7134                                  ;*TEST 33       FORMAT ERROR TEST - 16
7135                                  ;;*****************************************************************
7136  027664                          TST33:
7137  027664  000004                          SCOPE                   ;SCOPE CALL
7138  027666  000240                          NOP                     ;START OF TEST
7139  027670  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7140  027674  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7141  027700  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7142  027704  012737  000033  001226          MOV     #33,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7143  027712                          10$:
7144
7145                                  ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
7146  027712  012737  000000  001432          MOV     #0,RMDC0                ;CYLINDER = 0
7147  027720  012737  000000  001404          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
7148  027726  012737  010000  001430          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
7149  027734  012737  177376  001400          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
7150  027742  012737  102574  001402          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
7151  027750  012737  000062  001376          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
7152  027756  012737  064110  001174          MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
7153  027764  012737  000001  001176          MOV     #1,$TMP1
7154  027772  004737  036114                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
7155  027776                          20$:
7156
7157                                  ;PREPARE DEVICE FOR DATA TRANSFER
7158  027776  004737  033274                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7159  030002  154130                          .WORD   154130  ;TASK DESCRIPTOR
7160  030004  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
7161  030006  000240                          NOP                     ;RETURN HERE IF ERROR
7162  030010  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7163  030012  000137  030406                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7164  030016                          30$:
7165
7166                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
7167  030016  012737  000063  001376          MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
7168  030024  012702  001533                  MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
7169  030030  112722  000006                  MOVB    #RMDA,(R2)+
```

```
7170  030034  112722  000034          MOVB    #RMDC,(R2)+
7171  030040  112722  000032          MOVB    #RMOF,(R2)+
7172  030044  112722  000002          MOVB    #RMWC,(R2)+
7173  030050  112722  000004          MOVB    #RMBA,(R2)+
7174  030054  112722  000000          MOVB    #RMCS1,(R2)+
7175  030060  112722  000200          MOVB    #200,(R2)+
7176  030064  004737  037262          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7177  030070  000404                  BR      80$     ;GO TO 80$ IF NO ERROR
7178  030072  000240                  NOP                     ;RETURN HERE IF ERROR
7179  030074  104000                  ERROR                   ;ERROR DEFINED BY PUT SUB
7180  030076  000137  030406          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7181  030102                  80$:
7182
7183                          ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
7184  030102  004737  036726          JSR     PC,GETSTS
7185
7186                          ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
7187  030106  004737  037622          JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
7188  030112  004737  037012          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7189  030116  000404                  BR      90$     ;GO TO 90$ IF NO ERROR
7190  030120  000240                  NOP                     ;RETURN HERE IF ERROR
7191  030122  104000                  ERROR                   ;ERROR DEFINED BY GET SUB
7192  030124  000137  030406          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7193  030130                  90$:
7194
7195                          ;VERIFY RESULTS OF WRITE COMMAND
7196  030130  004737  040006          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7197  030134  000405                  BR      100$                    ;GO TO 100$ IF NO ERROR
7198  030136  000240                  NOP                     ;RETURN HERE IF ERROR
7199  030140  104000                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7200  030142  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7201  030144  000137  030406          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7202  030150                  100$:
7203  030150  004737  052312          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7204  030154  000405                  BR      110$                    ;GO TO 110$ IF NO ERROR
7205  030156  000240                  NOP                     ;RETURN HERE IF ERROR
7206  030160  104000                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7207  030162  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7208  030164  000137  030406          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7209  030170                  110$:
7210  030170  004737  040640          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7211  030174  000405                  BR      120$                    ;GO TO 120$ IF NO ERROR
7212  030176  000240                  NOP                     ;RETURN HERE IF ERROR
7213  030200  104000                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7214  030202  004736                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7215  030204  000137  030406          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7216  030210                  120$:
7217
7218
7219                          ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN AT 18 BIT FORMAT
7220  030210  012737  000000  001430          MOV     #0,RMOF0                ;SET FOR 18 BIT FORMAT
7221  030216  012737  000073  001376          MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
7222  030224  012737  103600  001402          MOV     #BUFTWO,RMBA0           ;CHANGE BUS ADDRESS
7223  030232  004737  037262                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7224  030236  000404                  BR      130$    ;GO TO 130$ IF NO ERROR
7225  030240  000240                  NOP                     ;RETURN HERE IF ERROR
```

```
7226  030242  104000                        ERROR                           ;ERROR DEFINED BY PUT SUB
7227  030244  000137  030406                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7228  030250                        130$:
7229
7230                                ;WAIT FOR READ TO COMPLETE AND GET STATUS
7231  030250  004737  037622                JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
7232  030254  004737  037012                JSR     PC,GET                  ;GO READ REGISTERS VIA GET SUB
7233  030260  000404                        BR      140$    ;GO TO 140$ IF NO ERROR
7234  030262  000240                        NOP                             ;RETURN HERE IF ERROR
7235  030264  104000                        ERROR                           ;ERROR DEFINED BY GET SUB
7236  030266  000137  030406                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7237  030272                        140$:
7238
7239                                ;VERIFY THE RESULTS OF READ OPERATION
7240  030272  004737  040006                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7241  030276  000405                        BR      150$                    ;GO TO 150$ IF NO ERROR
7242  030300  000240                        NOP                             ;RETURN HERE IF ERROR
7243  030302  104000                        ERROR                           ;ERROR # DEFINED BY PRIERR SUBROUTINE
7244  030304  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7245  030306  000137  030406                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7246  030312                        150$:
7247
7248                                ;VERIFY THAT FORMAT ERROR IS SET
7249  030312  032737  000020  001342        BIT     #FER,RMER1I             ;IS FORMAT ERROR SET??
7250  030320  001022                        BNE     160$                    ;YES!!
7251  030322  004737  052312                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7252  030326  000405                        BR      155$                    ;GO TO 155$ IF NO ERROR
7253  030330  000240                        NOP                             ;RETURN HERE IF ERROR
7254  030332  104000                        ERROR                           ;ERROR # DEFINED BY DTASTS SUBROUTINE
7255  030334  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7256  030336  000137  030406                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7257  030342  013737  001342  001142  155$:  MOV     RMER1I,$BDDAT           ;RECEIVED STATUS FOR TYPEOUT
7258  030350  013737  001342  001140        MOV     RMER1I,$GDDAT           ;EXPECTED STATUS
7259  030356  052737  000020  001140        BIS     #FER,$GDDAT
7260  030364  104343                        ERROR   343                     ;FORMAT ERROR NOT SET
7261  030366                        160$:
7262  030366  004737  040640                JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7263  030372  000405                        BR      170$                    ;GO TO 170$ IF NO ERROR
7264  030374  000240                        NOP                             ;RETURN HERE IF ERROR
7265  030376  104000                        ERROR                           ;ERROR # DEFINED BY SECERR SUBROUTINE
7266  030400  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7267  030402  000137  030406                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7268  030406                        170$:
7269  030406                        180$:
7270
7271                                ;;*****************************************************************
7272                                ;*TEST 34       FORMAT HCE, FIRST HEADER WORD
7273                                ;;*****************************************************************
7274  030406                        TST34:
7275  030406  000004                        SCOPE                           ;SCOPE CALL
7276  030410  000240                        NOP                             ;START OF TEST
7277  030412  012706  001100                MOV     #STACK,SP       ;INITIALIZE STAC  POINTER
7278  030416  013700  001276                MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7279  030422  013701  001446                MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7280  030426  012737  000034  001226        MOV     #34,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7281  030434  012737  000001  031412        MOV     #1,190$                 ;INIT BIT POSITION
```

```
7282  030442                                  10$:
7283
7284                                          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
7285  030442  012737  000000  001432                MOV     #0,RMOC0                ;CYLINDER = 0
7286  030450  012737  000000  001404                MOV     #0,RMDA0                ;TRACK = SECTOR = 0
7287  030456  012737  010000  001430                MOV     #FMT16,RMOF0            ;16 BIT FORMAT
7288  030464  012737  177376  001400                MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
7289  030472  012737  102574  001402                MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
7290  030500  012737  000062  001376                MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
7291  030506  012737  064110  001174                MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
7292  030514  012737  000001  001176                MOV     #1,$TMP1
7293  030522  004737  036114                      JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
7294  030526                                  20$:
7295
7296                                          ;PREPARE DEVICE FOR DATA TRANSFER
7297  030526  004737  033274                      JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7298  030532  154130                              .WORD   154130  ;TASK DESCRIPTOR
7299  030534  000404                              BR      30$             ;GO TO 30$ IF NO ERROR
7300  030536  000240                              NOP                     ;RETURN HERE IF ERROR
7301  030540  104000                              ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
7302  030542  000137  031410                      JMP     180$            ;GO TO 180$ IF ERROR WAS FOUND
7303  030546                                  30$:
7304
7305                                          ;COMPLEMENT DATA BIT IN FIRST HEADER WORD
7306  030546  033737  031412  102574                BIT     190$,BUFONE             ;IS BIT IN HEADER ON??
7307  030554  001404                              BEQ     31$                     ;NO!!
7308  030556  043737  031412  102574                BIC     190$,BUFONE             ;RESET BIT IN HEADER
7309  030564  000403                              BR      32$
7310  030566  053737  031412  102574  31$:        BIS     190$,BUFONE             ;SET BIT IN HEADER
7311
7312  030574                                  32$:
7313
7314                                          ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
7315  030574  012737  000063  001376                MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
7316  030602  012702  001533                      MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
7317  030606  112722  000005                      MOVB    #RMDA,(R2)+
7318  030612  112722  000034                      MOVB    #RMDC,(R2)+
7319  030616  112722  000032                      MOVB    #RMOF,(R2)+
7320  030622  112722  000002                      MOVB    #RMWC,(R2)+
7321  030626  112722  000004                      MOVB    #RMBA,(R2)+
7322  030632  112722  000000                      MOVB    #RMCS1,(R2)+
7323  030636  112722  000200                      MOVB    #200,(R2)+
7324  030642  004737  037262                      JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7325  030646  000404                              BR      80$     ;GO TO 80$ IF NO ERROR
7326  030650  000240                              NOP                     ;RETURN HERE IF ERROR
7327  030652  104000                              ERROR                   ;ERROR DEFINED BY PUT SUB
7328  030654  000137  031410                      JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7329  030660                                  80$:
7330
7331                                          ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
7332  030660  004737  036726                      JSR     PC,GETSTS
7333
7334                                          ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
7335  030664  004737  037622                      JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
7336  030670  004737  037012                      JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7337  030674  000404                              BR      90$     ;GO TO 90$ IF NO ERROR
```

```
7338  030676  000240                          NOP                     ;RETURN HERE IF ERROR
7339  030700  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
7340  030702  000137  031410                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7341  030706                          90$:
7342
7343                                  ;VERIFY RESULTS OF WRITE COMMAND
7344  030706  004737  040006                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7345  030712  000405                          BR      100$                    ;GO TO 100$ IF NO ERROR
7346  030714  000240                          NOP                     ;RETURN HERE IF ERROR
7347  030716  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7348  030720  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7349  030722  000137  031410                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7350  030726                          100$:
7351  030726  004737  052312                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7352  030732  000405                          BR      110$                    ;GO TO 110$ IF NO ERROR
7353  030734  000240                          NOP                     ;RETURN HERE IF ERROR
7354  030736  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7355  030740  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7356  030742  000137  031410                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7357  030746                          110$:
7358  030746  004737  040640                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7359  030752  000405                          BR      120$                    ;GO TO 120$ IF NO ERROR
7360  030754  000240                          NOP                     ;RETURN HERE IF ERROR
7361  030756  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7362  030760  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7363  030762  000137  031410                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7364  030766                          120$:
7365
7366
7367                                  ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
7368  030766  012737  000073  001376          MOV     #RH!GO,RMCS1O           ;READ HEADER & DATA COMMAND
7369  030774  012737  103600  001402          MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
7370  031002  004737  037262                  JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7371  031006  000404                          BR      130$    ;GO TO 130$ IF NO ERROR
7372  031010  000240                          NOP                     ;RETURN HERE IF ERROR
7373  031012  104000                          ERROR                   ;ERROR DEFINED BY PUT SUB
7374  031014  000137  031410                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7375  031020                          130$:
7376
7377                                  ;WAIT FOR READ TO COMPLETE AND GET STATUS
7378  031020  004737  037622                  JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
7379  031024  004737  037012                  JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7380  031030  000404                          BR      140$    ;GO TO 140$ IF NO ERROR
7381  031032  000240                          NOP                     ;RETURN HERE IF ERROR
7382  031034  104000                          ERROR                   ;ERROR DEFINED BY GET SUB
7383  031036  000137  031410                  JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7384  031042                          140$:
7385
7386                                  ;VERIFY THE RESULTS OF READ OPERATION
7387  031042  004737  040006                  JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7388  031046  000405                          BR      150$                    ;GO TO 150$ IF NO ERROR
7389  031050  000240                          NOP                     ;RETURN HERE IF ERROR
7390  031052  104000                          ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7391  031054  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7392  031056  000137  031410                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7393  031062                          150$:
```

```
7394  031062  032737  140000  031412          BIT     #MSE!USE,190$           ;IS THIS BSE ??
7395  031070  001033                          BNE     152$                    ;YES !!
7396  031072  032737  010000  031412          BIT     #FMT16,190$             ;IS THIS FER ??
7397  031100  001466                          BEQ     155$                    ;NO !!
7398
7399                                  ;VERIFY THAT FER IS SET
7400  031102  032737  000020  001342          BIT     #FER,RMER1I             ;IS FER SET ??
7401  031110  001122                          BNE     160$                    ;YES !!
7402  031112  004737  052312                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7403  031116  000405                          BR      151$                    ;GO TO 151$ IF NO ERROR
7404  031120  000240                          NOP                     ;RETURN HERE IF ERROR
7405  031122  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7406  031124  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7407  031126  000137  031410                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7408  031132  013737  001342  001142  151$:   MOV     RMER1I,$BDDAT           ;RECEIVED STATUS
7409  031140  013737  001342  001140          MOV     RMER1I,$GDDAT           ;EXPECTED STATUS
7410  031146  052737  000020  001140          BIS     #FER,$GDDAT
7411  031154  104343                          ERROR   343                     ;FORMAT ERROR NOT SET
7412  031156  000514                          BR      180$
7413  031160                          152$:
7414                                  ;VERIFY THAT BAD SECTOR ERROR IS SET
7415  031160  032737  100000  001370          BIT     #BSE,RMER2I             ;IS BSE SET ??
7416  031166  001073                          BNE     160$                    ;YES !!
7417  031170  004737  052312                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7418  031174  000405                          BR      153$                    ;GO TO 153$ IF NO ERROR
7419  031176  000240                          NOP                     ;RETURN HERE IF ERROR
7420  031200  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7421  031202  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7422  031204  000137  031410                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7423  031210                          153$:
7424  031210  013737  001370  001142          MOV     RMER2I,$BDDAT           ;RECEIVED STATUS
7425  031216  013737  001370  001140          MOV     RMER2I,$GDDAT           ;EXPECTED STATUS
7426  031224  052737  100000  001140          BIS     #BSE,$GDDAT
7427  031232  013737  102574  001174          MOV     BUFONE,$TMP0            ;WRITTEN HEADER
7428  031240  013737  103600  001176          MOV     BUFTWO,$TMP1            ;READ HEADER WORD
7429  031246  013737  031412  001200          MOV     190$,$TMP2              ;FAILING BIT POSITION
7430                                  ;       ERROR   345                     ;FAILED TO DETECT BSE
7431  031254  000455                          BR      180$                    ;SKIP REST OF TEST
7432  031256                          155$:
7433
7434                                  ;VERIFY THAT HEADER COMPARE ERROR IS SET
7435  031256  032737  000200  001342          BIT     #HCE,RMER1I             ;IS "HCE" SET??
7436  031264  001034                          BNE     160$                    ;YES!!
7437  031266  004737  052312                  JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7438  031272  000405                          BR      156$                    ;GO TO 156$ IF NO ERROR
7439  031274  000240                          NOP                     ;RETURN HERE IF ERROR
7440  031276  104000                          ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7441  031300  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7442  031302  000137  031410                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7443  031306  013737  0C1342  001142  156$:   MOV     RMER1I,$BDDAT           ;RECEIVED STATUS FOR TYPEOUT
7444  031314  013737  001342  001140          MOV     RMER1I,$GDDAT           ;EXPECTED STATUS
7445  031322  052737  000200  001140          BIS     #HCE,$GDDAT
7446  031330  013737  102574  001174          MOV     BUFONE,$TMP0            ;WRITTEN HEADER
7447  031336  013737  103600  001176          MOV     BUFTWO,$TMP1            ;READ HEADER WORD
7448  031344  013737  031412  001200          MOV     190$,$TMP2              ;FAILING BIT POSITION
7449  031352  104344                          ERROR   344                     ;FORMAT ERROR NOT SET
```

```
7450  031354  000415                          BR      180$
7451
7452  031356                                  160$:
7453
7454                                          ;CHECK FOR OTHER ERRORS
7455  031356  004737  040640                  JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7456  031362  000405                          BR      165$                    ;GO TO 165$ IF NO ERROR
7457  031364  000240                          NOP                     ;RETURN HERE IF ERROR
7458  031366  104000                          ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7459  031370  004736                          JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7460  031372  000137  031410                  JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7461  031376                                  165$:
7462
7463
7464
7465                                          ;ADVANCE THE BIT POSITION AND FORMAT AGAIN IF NOT DONE
7466  031376  006337  031412                  ASL     190$                    ;SHIFT TO NEXT BIT POSITION
7467  031402  001402                          BEQ     180$                    ;EXIT IF DONE
7468  031404  000137  030442                  JMP     10$                     ;GO DO NEXT SECTOR
7469  031410                                  180$:
7470  031410  000401                          BR      200$                    ;JUMP OVER STORAGE
7471  031412                                  190$:
7472  031412  000000                          .WORD                           ;STORAGE FOR BIT POSITION
7473  031414                                  200$:
7474
7475                                          ;;*****************************************************************
7476                                          ;*TEST 35       FORMAT HCE, SECOND HEADER WORD
7477                                          ;;*****************************************************************
7478  031414                                  TST35:
7479  031414  000004                          SCOPE                   ;SCOPE CALL
7480  031416  000240                          NOP                     ;START OF TEST
7481  031420  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
7482  031424  013700  001276                  MOV     $BASE,R0        ;R0=UNIBUS ADDRESS
7483  031430  013701  001446                  MOV     TSTQUE,R1       ;(R1) = DEVICE BEING TESTED
7484  031434  012737  000035  001226          MOV     #35,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
7485  031442  012737  000001  032224          MOV     #1,190$                 ;INIT BIT POSITION
7486  031450                                  10$:
7487
7488                                          ;SETUP PARAMETERS FOR GENERATING DATA BUFFER
7489  031450  012737  000000  001432          MOV     #0,RMDC0                ;CYLINDER = 0
7490  031456  012737  000000  001404          MOV     #0,RMDA0                ;TRACK = SECTOR = 0
7491  031464  012737  010000  001430          MOV     #FMT16,RMOF0            ;16 BIT FORMAT
7492  031472  012737  177376  001400          MOV     #<^C<2+256.>+1>,RMWC0   ;2 + 256 WORDS
7493  031500  012737  102574  001402          MOV     #BUFONE,RMBA0           ;DATA BUFFER ADDRESS
7494  031506  012737  000062  001376          MOV     #WH,RMCS10              ;WRITE HEADER AND DATA
7495  031514  012737  064110  001174          MOV     #ZEROS,$TMP0            ;USE ALL ZEROS DATA PATTERN
7496  031522  012737  000001  001176          MOV     #1,$TMP1
7497  031530  004737  036114                  JSR     PC,GENBUF               ;GO GENERATE DATA BUFFER
7498  031534                                  20$:
7499
7500                                          ;PREPARE DEVICE FOR DATA TRANSFER
7501  031534  004737  033274                  JSR     PC,TSTPRP       ;PREPARE DEVICE FOR TEST
7502  031540  154130                          .WORD   154130  ;TASK DESCRIPTOR
7503  031542  000404                          BR      30$                     ;GO TO 30$ IF NO ERROR
7504  031544  000240                          NOP                     ;RETURN HERE IF ERROR
7505  031546  104000                          ERROR                   ;ERROR # DEFINED BY TSTPRP SUBROUTINE
```

```
7506  031550  000137  032222            JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7507  031554                          30$:
7508
7509                                  ;COMPLEMENT DATA BIT IN SECOND HEADER WORD
7510  031554  033737  032224  102576        BIT     190$,BUFONE+2           ;IS BIT IN HEADER ON??
7511  031562  001404                        BEQ     31$                     ;NO!!
7512  031564  043737  032224  102576        BIC     190$,BUFONE+2           ;RESET BIT IN HEADER
7513  031572  000403                        BR      32$
7514  031574  053737  032224  102576  31$:  BIS     190$,BUFONE+2           ;SET BIT IN HEADER
7515
7516  031602                          32$:
7517
7518                                  ;SETUP AND EXECUTE WRITE HEADER AND DATA COMMAND
7519  031602  012737  000063  001376        MOV     #WH!GO,RMCS10           ;WRITE HEADER AND DATA
7520  031610  012702  001533                MOV     #PUTINX,R2              ;WRITE REGISTER INDEX TABLE
7521  031614  112722  000006                MOVB    #RMDA,(R2)+
7522  031620  112722  000034                MOVB    #RMDC,(R2)+
7523  031624  112722  000032                MOVB    #RMOF,(R2)+
7524  031630  112722  000002                MOVB    #RMWC,(R2)+
7525  031634  112722  000004                MOVB    #RMBA,(R2)+
7526  031640  112722  000000                MOVB    #RMCS1,(R2)+
7527  031644  112722  000200                MOVB    #200,(R2)+
7528  031650  004737  037262                JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7529  031654  000404                        BR      80$     ;GO TO 80$ IF NO ERROR
7530  031656  000240                        NOP                     ;RETURN HERE IF ERROR
7531  031660  104000                        ERROR                   ;ERROR DEFINED BY PUT SUB
7532  031662  000137  032222                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7533  031666                          80$:
7534
7535                                  ;SETUP INPUT REGISTER BUFFER FOR READING STATUS
7536  031666  004737  036726                JSR     PC,GETSTS
7537
7538                                  ;WAIT FOR WRITE COMMAND TO COMPLETE AND READ STATUS
7539  031672  004737  037622                JSR     PC,TIMOUT               ;WAIT FOR COMMAND TO COMPLETE
7540  031676  004737  037012                JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7541  031702  000404                        BR      90$     ;GO TO 90$ IF NO ERROR
7542  031704  000240                        NOP                     ;RETURN HERE IF ERROR
7543  031706  104000                        ERROR                   ;ERROR DEFINED BY GET SUB
7544  031710  000137  032222                JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7545  031714                          90$:
7546
7547                                  ;VERIFY RESULTS OF WRITE COMMAND
7548  031714  004737  040006                JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7549  031720  000405                        BR      100$                    ;GO TO 100$ IF NO ERROR
7550  031722  000240                        NOP                     ;RETURN HERE IF ERROR
7551  031724  104000                        ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7552  031726  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7553  031730  000137  032222                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7554  031734                          100$:
7555  031734  004737  052312                JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7556  031740  000405                        BR      110$                    ;GO TO 110$ IF NO ERROR
7557  031742  000240                        NOP                     ;RETURN HERE IF ERROR
7558  031744  104000                        ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7559  031746  004736                        JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7560  031750  000137  032222                JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7561  031754                          110$:
```

```
7562  031754  004737  040640                          JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7563  031760  000405                                  BR      120$                    ;GO TO 120$ IF NO ERROR
7564  031762  000240                                  NOP                     ;RETURN HERE IF ERROR
7565  031764  104000                                  ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7566  031766  004736                                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7567  031770  000137  032222                          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7568  031774                                  120$:
7569
7570
7571                                          ;READ HEADER AND DATA FOR SECTOR JUST WRITTEN
7572  031774  012737  000073  001376                  MOV     #RH!GO,RMCS10           ;READ HEADER & DATA COMMAND
7573  032002  012737  103600  001402                  MOV     #BUFTWO,RMBAO           ;CHANGE BUS ADDRESS
7574  032010  004737  037262                          JSR     PC,PUT          ;GO WRITE REGISTERS VIA PUT SUB
7575  032014  000404                                  BR      130$    ;GO TO 130$ IF NO ERROR
7576  032016  000240                                  NOP                     ;RETURN HERE IF ERROR
7577  032020  104000                                  ERROR                   ;ERROR DEFINED BY PUT SUB
7578  032022  000137  032222                          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7579  032026                                  130$:
7580
7581                                          ;WAIT FOR READ TO COMPLETE AND GET STATUS
7582  032026  004737  037622                          JSR     PC,TIMOUT               ;WAIT FOR READ TO COMPLETE
7583  032032  004737  037012                          JSR     PC,GET          ;GO READ REGISTERS VIA GET SUB
7584  032036  000404                                  BR      140$    ;GO TO 140$ IF NO ERROR
7585  032040  000240                                  NOP                     ;RETURN HERE IF ERROR
7586  032042  104000                                  ERROR                   ;ERROR DEFINED BY GET SUB
7587  032044  000137  032222                          JMP     180$    ;GO TO 180$ IF ERROR WAS FOUND
7588  032050                                  140$:
7589
7590                                          ;VERIFY THE RESULTS OF READ OPERATION
7591  032050  004737  040006                          JSR     PC,PRIERR       ;GO CHECK FOR PRIMARY ERRORS
7592  032054  000405                                  BR      150$                    ;GO TO 150$ IF NO ERROR
7593  032056  000240                                  NOP                     ;RETURN HERE IF ERROR
7594  032060  104000                                  ERROR                   ;ERROR # DEFINED BY PRIERR SUBROUTINE
7595  032062  004736                                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7596  032064  000137  032222                          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7597  032070                                  150$:
7598
7599                                          ;VERIFY THAT HEADER COMPARE ERROR IS SET
7600  032070  032737  000200  001342                  BIT     #HCE,RMER1I                     ;IS "HCE" SET??
7601  032076  001034                                  BNE     160$                    ;YES!!
7602  032100  004737  052312                          JSR     PC,DTASTS       ;GO VERIFY RESULTS OF DATA TRANSFER
7603  032104  000405                                  BR      155$                    ;GO TO 155$ IF NO ERROR
7604  032106  000240                                  NOP                     ;RETURN HERE IF ERROR
7605  032110  104000                                  ERROR                   ;ERROR # DEFINED BY DTASTS SUBROUTINE
7606  032112  004736                                  JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7607  032114  000137  032222                          JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7608  032120  013737  001342  001142  155$:           MOV     RMER1I,$BDDAT           ;RECEIVED STATUS FOR TYPEOUT
7609  032126  013737  001342  001140                  MOV     RMER1I,$GDDAT           ;EXPECTED STATUS
7610  032134  052737  000200  001140                  BIS     #HCE,$GDDAT
7611  032142  013737  102576  001174                  MOV     BUFONE+2,$TMP0          ;WRITTEN HEADER
7612  032150  013737  103602  001176                  MOV     BUFTWO+2,$TMP1          ;READ HEADER WORD
7613  032156  013737  032224  001200                  MOV     190$,$TMP2              ;FAILING BIT POSITION
7614  032164  104344                                  ERROR   344                     ;FORMAT ERROR NOT SET
7615  032166  000415                                  BR      180$
7616
7617  032170                                  160$:
```

```
7618
7619                                    ;CHECK FOR OTHER ERRORS
7620  032170  004737  040640                    JSR     PC,SECERR       ;GO CHECK FOR SECONDARY ERRORS
7621  032174  000405                            BR      165$                    ;GO TO 165$ IF NO ERROR
7622  032176  000240                            NOP                     ;RETURN HERE IF ERROR
7623  032200  104000                            ERROR                   ;ERROR # DEFINED BY SECERR SUBROUTINE
7624  032202  004736                            JSR     PC,@(SP)+       ;GO BACK FOR MORE ERROR CHECKS
7625  032204  000137  032222                    JMP     180$                    ;GO TO 180$ IF ERROR WAS FOUND
7626  032210                                165$:
7627
7628
7629                                    ;ADVANCE THE BIT POSITION AND FORMAT NEXT SECTOR IF NOT DONE
7630  032210  006337  032224                    ASL     190$                    ;SHIFT TO NEXT BIT POSITION
7631  032214  001402                            BEQ     180$                    ;EXIT IF DONE
7632  032216  000137  031450                    JMP     10$                     ;GO DO NEXT SECTOR
7633  032222                                180$:
7634  032222  000401                            BR      200$                    ;JUMP OVER STORAGE
7635  032224                                190$:
7636  032224  000000                            .WORD                   ;STORAGE FOR BIT POSITION
7637  032226                                200$:
7638
7639                                    ;PUT NEWTEST HERE
```

```
7640  032226                                  $EOSP:
7641  032226  000240                                  NOP
7642  032230  013700  001446                          MOV     TSTQUE,R0
7643  032234  062700  000002                          ADD     #2,R0
7644  032240  010037  001446                          MOV     R0,TSTQUE
7645  032244  005710                                  TST     (R0)
7646  032246  001402                                  BEQ     1$
7647  032250  000137  006764                          JMP     READY
7648  03225^  012737  001450  001446  1$:             MOV     #TSTQUE+2,TSTQUE
7649                                    .SBTTL  END OF PASS ROUTINE
7650
7651                                    ;;**********************************************************************
7652                                    ;*INCREMENT THE PASS NUMBER ($PASS)
7653                                    ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
7654                                    ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
7655                                    ;*IF THERES A MONITOR GO TO IT
7656                                    ;*IF THERE ISN'T JUMP TO READY
7657
7658  032262                            $EOP:
7659  032262  000240                                  NOP
7660  032264  005037  001116                          CLR     $TSTNM          ;;ZERO THE TEST NUMBER
7661  032270  005037  001206                          CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
7662  032274  005237  001230                          INC     $PASS           ;;INCREMENT THE PASS NUMBER
7663  032300  042737  100000  001230                  BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
7664  032306  005327                                  DEC     (PC)+           ;;LOOP?
7665  032310  000001                    $EOPCT: .WORD   1
7666  032312  003063                                  BGT     $DOAGN          ;;YES
7667  032314  012737                                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
7668  032316  000001                    $ENDCT: .WORD   1
7669  032320  032310                                  $EOPCT
7670  032322  104401  032330                        TYPE    ,65$            ;;TYPE ASCIZ STRING
7671  032326  000407                                  BR      64$             ;;GET OVER THE ASCIZ
7672                                    ;;65$:  .ASCIZ  <12><15>/END PASS #/
7673  032346                            64$:
7674  032346  013746  001230                          MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
7675                                                                          ;;TYPE PASS NUMBER
7676  032352  104405                                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
7677  032354  104401  032362                        TYPE    ,67$            ;;TYPE ASCIZ STRING
7678  032360  000421                                  BR      66$             ;;GET OVER THE ASCIZ
7679                                    ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
7680  032424                            66$:
7681  032424  013746  001126                          MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
7682                                                                          ;;TOTAL NUMBER OF ERRORS
7683  032430  104405                                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
7684  032432  104401  001217                          TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
7685  032436  005037  001126                          CLR     $ERTTL          ;;CLEAR ERROR TOTAL
7686  032442  013700  000042            $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
7687  032446  001405                                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
7688  032450  000005                                  RESET                   ;;CLEAR THE WORLD
7689  032452  004710                    $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
7690  032454  000240                                  NOP                     ;;SAVE ROOM
7691  032456  000240                                  NOP                     ;;FOR
7692  032460  000240                                  NOP                     ;;ACT11
7693  032462                            $DOAGN:
7694  032462  000137                                  JMP     @(PC)+          ;;RETURN
7695  032464  006764                    $RTNAD: .WORD   READY
```

```
 7696  032466     377     377     000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
 7697          032472                          .EVEN
```

```
                                        .SBTTL  SUBROUTINES

                                        ;;*****************************************************************
                                        .SBTTL  ERROR TYPEOUT ROUTINE

                                        ;*THE ERROR TYPEOUT ROUTINE ASSEMBLES AND PRINTS INFORMATION
                                        ;*REGARDING THE DETECTION OF AN ERROR AS FOLLOWS:
                                        ;*
                                        ;*      .UNIT NUMBER, TEST NUMBER, ERROR NUMBER AND PROGRAM COUNTER ARE
                                        ;*PRINTED ON THE FIRST LINE;
                                        ;*      .ERROR MESSAGE IS ASSEMBLED, FORMATTED AND PRINTED ON
                                        ;*ONE OR MORE SUCCEEDING LINES;
                                        ;*      .PAIRED LINES OF ERROR HEADERS AND ERROR DATA
                                        ;*ARE PRINTED AFTER THE ERROR MESSAGE.

032472                                  ERRTYP:
032472  104414                                  SAVREG
032474  032777  020000  146452                  BIT     #SW13,@SWR      ;INHIBIT TYPEOUTS??
032502  001402                                  BEQ     1$              ;NO!!
032504  000137  033222                          JMP     21$             ;YES!!
                                        ;TYPE UNIT NUMBER, TEST NUMBER, ERROR NUMBER, AND PROGRAM COUNTER
032510  104401  001217                  1$:     TYPE    ,$CRLF
032514  104401  033236                          TYPE    ,ERTY00         ;TYPE "UNT#"
032520  013746  001234                          MOV     $UNIT,-(SP)     ;;SAVE $UNIT FOR TYPEOUT
                                                                        ;;TYPE UNIT NUMBER
032524  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
032526     003                                  .BYTE   3               ;;TYPE 3 DIGIT(S)
032527     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
032530  005037  033226                          CLR     TSTNMB          ;LOAD TEST NUMBER FOR
032534  013737  001226  033226                  MOV     $TESTN,TSTNMB
032542  104401  033244                          TYPE    ,ERTY01         ;TYPE "TST#"
032546  013746  033226                          MOV     TSTNMB,-(SP)    ;;SAVE TSTNMB FOR TYPEOUT
                                                                        ;;TYPE TEST NUMBER
032552  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
032554     003                                  .BYTE   3               ;;TYPE 3 DIGIT(S)
032555     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
032556  005037  033230                          CLR     ERRNMB          ;LOAD ERROR NUMBER FOR
032562  113737  001130  033230                  MOVB    $ITEMB,ERRNMB   ;TYPEOUT
032570  001406                                  BEQ     2$              ;SKIP IF NO ERROR CALLED
032572  104401  033254                          TYPE    ,ERTY02         ;TYPE "ERR#"
032576  013746  033230                          MOV     ERRNMB,-(SP)    ;;SAVE ERRNMB FOR TYPEOUT
                                                                        ;;TYPE ERROR NUMBER
032602  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
032604     003                                  .BYTE   3               ;;TYPE 3 DIGIT(S)
032605     000                                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
032606  104401  033263          2$:     TYPE    ,ERTY03                 ;TYPE "PC="
032612  013746  001132                          MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                        ;;TYPE PROGRAM COUNTER
032616  104403                                  TYPOS                   ;;GO TYPE--OCTAL ASCII
032620     006                                  .BYTE   6               ;;TYPE 6 DIGIT(S)
032621     001                                  .BYTE   1               ;;TYPE LEADING ZEROS
                                        ;GENERATE POINTER TO ERROR TABLE UNLESS ERROR NUMBER IS 0
032622  005737  033230          3$:     TST     ERRNMB                  ;WAS AN ERROR CALLED?
032626  001575                          BEQ     21$                     ;NO!!
032630  104401  001217                  TYPE    ,$CRLF                  ;YES-TYPE CRLF
032634  105037  033234                  CLRB    BOTFLG                  ;CLEAR BOT FLAG
```

```
7754  032640  105037  033235            CLRB    CHRCNT          ;CLEAR CHARACTER COUNTER
7755  032644  013700  033230            MOV     ERRNMB,R0       ;R0 POINTS TO FIRST OF
7756  032650  006300                    ASL     R0              ;FOUR ENTRIES IN ERROR
7757  032652  006300                    ASL     R0              ;TABLE
7758  032654  006300                    ASL     R0
7759  032656  062700  001552            ADD     #$ERRTB-8.,R0
7760  032662  011001                    MOV     (R0),R1         ;R1 POINTS TO ERROR MESSAGE
7761                                                            ;TABLE
7762  032664  001507                    BEQ     13$             ;BRANCH IF NO ERROR MESSAGE
7763                            ;TYPE THE ERROR MESSAGE
7764  032666  012102            4$:     MOV     (R1)+,R2        ;R2=ADDRESS OF MESSAGE STRING
7765  032670  001505                    BEQ     12$             ;BRANCH IF END OF MESSAGE
7766  032672  010237  033040            MOV     R2,11$          ;LOAD ADDRESS OF STRING
7767  032676  005037  033232            CLR     BOTADR          ;CLEAR BOT ADDRESS
7768  032702  112203            5$:     MOVB    (R2)+,R3        ;END OF STRING??
7769  032704  001454                    BEQ     10$             ;YES!!
7770  032706  122703  000015            CMPB    #CR,R3          ;CARRIAGE RETURN??
7771  032712  001003                    BNE     6$              ;NO!!
7772  032714  105037  033235            CLRB    CHRCNT          ;YES-CLEAR CHAR COUNT
7773  032720  000770                    BR      5$              ;GET NEXT CHARACTER
7774  032722  122703  000012    6$:     CMPB    #LF,R3          ;LINE FEED??
7775  032726  001765                    BEQ     5$              ;YES-GET NEXT CHARACTER
7776  032730  122703  000011            CMPB    #HT,R3          ;HORIZONTAL TAB??
7777  032734  001007                    BNE     8$              ;NO!!
7778  032736  105237  033235    7$:     INCB    CHRCNT          ;ADJUST CHARACTER COUNT
7779  032742  132737  000007  033235    BITB    #7,CHRCNT
7780  032750  001372                    BNE     7$
7781  032752  000407                    BR      9$
7782  032754  105237  033235    8$:     INCB    CHRCNT          ;INCREMENT CHARACTER COUNT
7783  032760  122703  000040            CMPB    #' ,R3          ;SPACE??
7784  032764  001002                    BNE     9$              ;NO!!
7785  032766  010237  033232            MOV     R2,BOTADR       ;SAVE ADDRESS OF SPACE
7786  032772  122737  000100  033235    CMPB    #64.,CHRCNT     ;END OF LINE??
7787  033000  103340            9$:     BHIS    5$              ;NO!!
7788  033002  013704  033232            MOV     BOTADR,R4       ;GET ADDRESS OF LAST SPACE
7789  033006  001007                    BNE     90$             ;BRANCH IF SPACE DETECTED
7790  033010  104401  001217            TYPE    ,$CRLF          ;TYPE CRLF
7791  033014  105037  033235            CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
7792  033020  013702  033040            MOV     11$,R2          ;SET UP R2 FOR TESTING
7793  033024  000726                    BR      5$
7794  033026  105044            90$:    CLRB    -(R4)           ;REPLACE SPACE
7795  033030  112737  177777  033234    MOVB    #-1,BOTFLG      ;SET BOT FLAG
7796  033036  104401            10$:    TYPE                    ;TYPE ERROR MESSAGE STRING
7797  033040  000000            11$:    .WORD                   ;STRING ADDRESS GOES HERE
7798  033042  105737  033234            TSTB    BOTFLG          ;WAS STRING TRUNCATED??
7799  033046  001707                    BEQ     4$              ;NO!!
7800  033050  104401  001217            TYPE    ,$CRLF          ;YES-TYPE CRLF
7801  033054  105037  033234            CLRB    BOTFLG          ;CLEAR BOT FLAG
7802  033060  105037  033235            CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
7803  033064  013702  033232            MOV     BOTADR,R2       ;SETUP R2 FOR TESTING
7804  033070  010237  033040            MOV     R2,11$          ;SETUP 11$ FOR TYPING
7805  033074  112742  000040            MOVB    #' ,-(R2)       ;RESTORE SPACE
7806  033100  105722                    TSTB    (R2)+           ;RESTORE R2
7807  033102  000677                    BR      5$              ;TYPE REST OF STRING
7808  033104                    12$:
7809                            ;TYPE ERROR HEADER AND ERROR DATA
```

```
7810  033104
7811  033104  016001  000002          13$:    MOV     2(R0),R1        ;R1 POINTS TO ERROR HEADER TABLE
7812  033110  001444                          BEQ     21$             ;BRANCH IF NO HEADER
7813  033112  104401  001217                  TYPE    ,$CRLF          ;(ASSUME NO DATA)
7814  033116  016002  000004                  MOV     4(R0),R2        ;R2 POINTS TO DATA ADDRESS TABLE
7815  033122  016003  000006                  MOV     6(R0),R3        ;R3 POINTS TO FORMAT TABLE
7816  033126  012137  033136          14$:    MOV     (R1)+,15$       ;PUT HEADER ADDRESS FOR TYPE
7817  033132  001433                          BEQ     21$             ;BRANCH IF END OF HEADERS
7818                                                                  ;(ASSUME END OF DATA)
7819  033134  104401                          TYPE
7820  033136  000000          15$:    .WORD   0               ;HEADER ADDRESS GOES HERE
7821  033140  104401  001217                  TYPE    ,$CRLF
7822  033144  005702                          TST     R2              ;DATA WITH HEADER??
7823  033146  001767                          BEQ     14$             ;NO!!
7824  033150  012204                          MOV     (R2)+,R4        ;R4 POINTS TO DATA ADDRESS
7825  033152  012305                          MOV     (R3)+,R5        ;R5 POINTS TO FORMAT
7826  033154  105725          16$:    TSTB    (R5)+           ;WHAT KIND OF DATA??
7827  033156  100407                          BMI     18$             ;BINARY
7828  033160  001403                          BEQ     17$             ;OCTAL
7829  033162  013446                          MOV     @(R4)+,-(SP)    ;DECIMAL
7830  033164  104405                          TYPDS
7831  033166  000405                          BR      19$
7832  033170  013446          17$:    MOV     @(R4)+,-(SP)
7833  033172  104402                          TYPOC
7834  033174  000402                          BR      19$
7835  033176  013446          18$:    MOV     @(R4)+,-(SP)
7836  033200  104406                          TYPBN
7837  033202  005714          19$:    TST     (R4)            ;MORE DATA??
7838  033204  001403                          BEQ     20$             ;NO!!
7839  033206  104401  033271                  TYPE    ,ERTY04         ;YES-TYPE 2 SPACES
7840  033212  000760                          BR      16$             ;AND CONTINUE
7841  033214  104401  001217          20$:    TYPE    ,$CRLF          ;TYPE ONE BLANK LINE
7842  033220  000742                          BR      14$             ;BEFORE NEXT HEADER
7843  033222  104415          21$:    RESREG
7844  033224  000207                          RTS     PC
7845
7846  033226  000000          TSTNMB: .WORD   0               ;TEST NUMBER
7847  033230  000000          ERRNMB: .WORD   0               ;ERROR NUMBER
7848  033232  000000          BOTADR: .WORD   0               ;BEGINNING OF TEXT ADDRESS
7849  033234     000          BOTFLG: .BYTE   0               ;BOT FLAG
7850  033235     000          CHRCNT: .BYTE   0               ;CHARACTER COUNT
7851
7852  033236  047125  052111  000043  ERTY00: .ASCIZ  @UNIT#@
7853  033244  020054  042524  052123  ERTY01: .ASCIZ  @, TEST#@
7854  033252  000043
7855  033254  020054  051105  021522  ERTY02: .ASCIZ  @, ERR#@
7856  033262     000
7857  033263     054  050040  036503  ERTY03: .ASCIZ  @, PC=@
7858  033270     000
7859  033271     040  000040          ERTY04: .ASCIZ  @ @
7860                                  .EVEN
7861
```

```
                                .SBTTL  TEST PREPARATION MODULE

                                ; THIS MODULE PREPARES THE RM03 SUBSYSTEM FOR THE EXECUTION OF A TEST,
                                ; REPORTING AN ERROR TO THE USER IF AN ERROR IS  DETECTED.  THE USER
                                ; SPECIFIES TASKS TO BE PERFORMED, WHICH THE MODULE EXECUTES
                                ; USING SUBROUTINES.

                                ;CALL:
                                ;       JSR     PC,TSTPRP
                                ;       .WORD                   TASK/VERIFY DESCRIPTOR
                                ;       BR      ??              RETURN HERE IF NO ERROR
                                ;       NOP                     RETURN HERE IF ERROR
                                ;       ERROR                   ERROR DEFINED BY MODULE

                                ;TASK/VERIFY DESCRIPTOR
                                ;       BIT 15 = 1      SELECT DEVICE AND VERIFY DEVICE IS AVAILABLE
                                ;       BIT 14 = 1      CLEAR CONTROLLER AND SELECT DEVICE
                                ;       BIT 13          (RESERVED FOR DRIVE CLEAR)
                                ;       BIT 12 = 1      PACK ACKNOWLEDGE IF VOLUME NOT VALID
                                ;       BIT 11 = 1      RECALIBRATE IF POSITIONING IN PROGRESS
                                ;       BIT 10
                                ;       BIT 9
                                ;       BIT 8
                                ;       BIT 7
                                ;       BIT 6  = 1      VERIFY CONTROLLER CLEAR OPERATION
                                ;       BIT 5           (RESERVED FOR DRIVE CLEAR)
                                ;       BIT 4  = 1      VERIFY PACK ACKNOWLEDGE
                                ;       BIT 3  = 1      VERIFY RECALIBRATION
                                ;       BIT 2
                                ;       BIT 1
                                ;       BIT 0

033274                          TSTPRP:

                                ;STORE TASK DESCRIPTOR AND CLEAR USER'S ERROR CALL
033274  017637  000000  034206          MOV     @(SP),50O$      ;STORE DESCRIPTOR
033302  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
033306  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR CALL
033312  162716  000004                  SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
                                ;;*****************************************************************
                                ;SELECT DEVICE AND VERIFY DEVICE AVAILABLE IF BIT 15 SET IN TASK
033316  032737  100000  034206          BIT     #BIT15,50O$     ;SELECT DEVICE??
033324  001414                          BEQ     30$             ;NO!!
033326  004737  041502                  JSR     PC,DEVSEL       ;GO SELECT DEVICE
033332  000411                          BR      30$             ;NO ERROR - CONTINUE
033334  000401                          BR      20$
033336  000000                  10$:    .WORD                   ;ERROR NUMBER FROM DEVSEL
033340  062716  000004          20$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
033344  113776  033336  000000          MOVB    10$,@(SP)
033352  000137  034174                  JMP     400$
033356                          30$:

                                ;;*****************************************************************
                                ;CLEAR CONTROLLER IF BIT 14 IS SET IN TASK
033356  032737  040000  034206          BIT     #BIT14,50O$     ;CLEAR CONTROLLER??
033364  001453                          BEQ     120$            ;NO!!
```

```
7918  033366  004737  046154          JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
7919  033372  000411                  BR      60$             ;CONTINUE - NO ERROR
7920  033374  000401                  BR      50$
7921  033376  000000          40$:    .WORD                   ;ERROR NUMBER FROM CNTCLR
7922  033400  062716  000004  50$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
7923  033404  113776  033376  000000  MOVB    40$,@(SP)
7924  033412  000137  034174          JMP     400$
7925  033416                  60$:
7926                          ;;**************************************************************
7927                          ;VERIFY CONTROLLER CLEAR IF BIT6 SET IN TASK
7928  033416  032737  000100  034206  BIT     #BIT6,500$      ;VERIFY??
7929  033424  001433                  BEQ     120$            ;NO!!
7930  033426  004737  036726          JSR     PC,GETSTS       ;SETUP INDEX TABLE
7931  033432  004737  037012          JSR     PC,GET          ;GO GET STATUS
7932  033436  000411                  BR      90$             ;NO ERROR GETTING STATUS
7933  033440  000401                  BR      80$
7934  033442  000000          70$:    .WORD                   ;ERROR FROM GETTING STATUS
7935  033444  062716  000004  80$:    ADD     #4,(SP)         ;TRANSFER ERROR TO USER
7936  033450  113776  033442  000000  MOVB    70$,@(SP)
7937  033456  000137  034174          JMP     400$
7938  033462  004737  046272  90$:    JSR     PC,CLRSTS       ;GO VERIFY STATUS CLEAR
7939  033466  000412                  BR      120$            ;NO ERROR IN CLEAR
7940  033470  000401                  BR      110$
7941  033472  000000          100$:   .WORD                   ;ERROR IN STATUS CLEAR
7942  033474  005726          110$:   TST     (SP)+           ;STRIP RETURN ADDRESS TO
7943  033476  062716  000004          ADD     #4,(SP)         ;SUBROUTINE AND TRANSFER
7944  033502  113776  033472  000000  MOVB    100$,@(SP)      ;ERROR TO USER
7945  033510  000137  034174          JMP     400$
7946  033514                  120$:
7947
7948                          ;;**************************************************************
7949                          ;EXECUTE PACK ACKNOWLEDGE IF BIT12 SET IN TASK AND VOLUME IS
7950                          ;NOT VALID
7951  033514  032737  010000  034206          BIT     #BIT12,500$     ;PACK ACKNOWLEDGE??
7952  033522  001511                  BEQ     240$            ;NO!!
7953  033524  112737  000012  001504  MOVB    #RMDS,GETINX    ;GET RMDS
7954  033532  112737  000200  001505  MOVB    #200,GETINX+1
7955  033540  004737  037012          JSR     PC,GET
7956  033544  000411                  BR      150$            ;NO ERROR GETTING RMDS
7957  033546  000401                  BR      140$
7958  033550  000000          130$:   .WORD
7959  033552  062716  000004  140$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
7960  033556  113776  033550  000000  MOVB    130$,@(SP)
7961  033564  000137  034174          JMP     400$
7962  033570  032737  000100  001340  150$:   BIT     #VV,RMDSI       ;IS VOLUME VALID??
7963  033576  001063                  BNE     240$            ;YES!!
7964  033600  005037  001472          CLR     MEDENB          ;CLEAR MEDIA ENABLE
7965  033604  012737  000023  001376  MOV     #PAKACK!GO,RMCS1        ;LOAD PACK ACK COMMAND
7966  033612  112737  000000  001533  MOVB    #RMCS1,PUTINX   ;SETUP REGISTER INDEX TABLE
7967  033620  112737  000200  001534  MOVB    #200,PUTINX+1
7968  033626  004737  037262          JSR     PC,PUT          ;GO WRITE COMMAND
7969  033632  000410                  BR      180$            ;NO ERROR LOADING REGISTER
7970  033634  000401                  BR      170$
7971  033636  000000          160$:   .WORD                   ;ERROR FROM PUT SUB
7972  033640  062716  000004  170$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
7973  033644  113776  033636  000000  MOVB    160$,@(SP)
```

```
 7974  033652  000550                          BR      400$
 7975  033654                                  180$:
 7976
 7977                                          ;;*****************************************************************
 7978                                          ;VERIFY PACK ACKNOWLEDGE IF #BIT4 SET IN TASK
 7979  033654  032737  000020  034206          BIT     #BIT4,500$      ;VERIFY PACK ACKNOWLEDGE??
 7980  033662  001431                          BEQ     240$            ;NO!!
 7981  033664  004737  036726                  JSR     PC,GETSTS       ;SETUP FOR STATUS
 7982  033670  004737  037012                  JSR     PC,GET          ;GO GET STATUS
 7983  033674  000410                          BR      210$            ;NO ERROR GETTING STATUS
 7984  033676  000401                          BR      200$
 7985  033700  000000                          190$:   .WORD                   ;ERROR FROM GET SUB
 7986  033702  062716  000004                  200$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 7987  033706  113776  033700  000000                  MOVB    190$,@(SP)
 7988  033714  000527                          BR      400$
 7989  033716  004737  047152                  210$:   JSR     PC,ACKSTS       ;GO CHECK ACKNOWLEDGE
 7990  033722  000411                          BR      240$            ;NO ERROR
 7991  033724  000401                          BR      230$
 7992  033726  000000                          220$:   .WORD                   ;PACK ACKNOWLEDGE ERROR
 7993  033730  005726                          230$:   TST     (SP)+           ;STRIP RETURN TO SUB AND
 7994  033732  062716  000004                          ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 7995  033736  113776  033774  000000                  MOVB    220$,@(SP)
 7996  033744  000513                          BR      400$
 7997  033746                                  240$:
 7998
 7999                                          ;;*****************************************************************
 8000                                          ;RECALIBRATE DRIVE IF BIT 11 SET IN TASK AND PIP IS ACTIVE
 8001  033746  032737  004000  034206          BIT     #BIT11,500$     ;RECALIBRATE??
 8002  033754  001513                          BEQ     410$            ;NO!!
 8003  033756  112737  000012  001504          MOVB    #RMDS,GETINX    ;LOAD REGISTER INDEX TABLE
 8004  033764  112737  000200  001505          MOVB    #200,GETINX+1
 8005  033772  004737  037012                  JSR     PC,GET          ;GO GET RMDS
 8006  033776  000410                          BR      270$            ;NO ERROR GETTING RMDS
 8007  034000  000401                          BR      260$
 8008  034002  000000                          250$:   .WORD                   ;ERROR FROM GET SUB
 8009  034004  062716  000004                  260$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 8010  034010  113776  034002  000000                  MOVB    250$,@(SP)
 8011  034016  000466                          BR      400$
 8012  034020  032737  020000  001340          270$:   BIT     #PIP,RMDSI      ;IS PIP ACTIVE??
 8013  034026  001466                          BEQ     410$            ;NO!!
 8014  034030  012737  000007  001376          MOV     #RECAL!GO,RMCS10;LOAD RECALIBRATE COMMAND
 8015  034036  112737  000000  001533          MOVB    #RMCS1,PUTINX   ;AND REGISTER INDEX
 8016  034044  112737  000200  001534          MOVB    #200,PUTINX+1
 8017  034052  004737  037262                  JSR     PC,PUT          ;GO ISSUE RECALIBRATE
 8018  034056  000410                          BR      300$            ;NO ERROR
 8019  034060  000401                          BR      290$
 8020  034062  000000                          280$:   .WORD                   ;ERROR IN REGISTER TRANSFER
 8021  034064  062716  000004                  290$:   ADD     #4,(SP)         ;TRANSFER ERROR TO USER
 8022  034070  113776  034062  000000                  MOVB    280$,@(SP)
 8023  034076  000436                          BR      400$
 8024  034100  004737  036726                  300$:   JSR     PC,GETSTS       ;SETUP FOR STATUS
 8025  034104  004737  037622                  JSR     PC,TIMOUT       ;WAIT FOR COMPLETION
 8026
 8027                                          ;;*****************************************************************
 8028                                          ;VERIFY RECALIBRATE IF BIT 3 SET IN TASK
 8029  034110  032737  000010  034206          BIT     #BIT3,500$      ;VERIFY RECALIBRATE??
```

```
8030  034116  001432                        BEQ     410$            ;NO!!
8031  034120  004737  037012                JSR     PC,GET          ;GO GET STATUS
8032  034124  000410                        BR      330$            ;NO ERROR GETTING STATUS
8033  034126  000401                        BR      320$
8034  034130  000000              310$:     .WORD                   ;ERROR FROM GET
8035  034132  062716  000004      320$:     ADD     #4,(SP)         ;TRANSFER ERROR TO USER
8036  034136  113776  034130  000000        MOVB    310$,@(SP)
8037  034144  000413                        BR      400$
8038  034146  004737  047746      330$:     JSR     PC,RCLSTS       ;GO CHECK RECALIBRATE
8039  034152  000414                        BR      410$            ;NO ERROR DURING RECALIBRATE
8040  034154  000401                        BR      350$
8041  034156  000000              340$:     .WORD                   ;ERROR DURING RECALIBRATE
8042  034160  005726              350$:     TST     (SP)+           ;STRIP RETURN TO SUB AND
8043  034162  062716  000004              ADD     #4,(SP)         ;TRANSFER ERROR TO USER
8044  034166  113776  034156  000000        MOVB    340$,@(SP)
8045  034174  162716  000002      400$:     SUB     #2,(SP)         ;MOVE SP BACK BEFORE ERROR
8046  034200  000240                        NOP
8047  034202  000240                        NOP
8048  034204  000207              410$:     RTS     PC              ;RETURN TO USER
8049
8050  034206  000000              500$:     .WORD                   ;TASK/VERIFY DESCRIPTOR
```

```
 8051                           .SBTTL  BAD SECTOR MODULE
 8052
 8053                           ;THE BAD SECTOR MODULE PERFORMS TWO MAJOR FUNCTIONS, I.E.,
 8054                           ;       RECOVER THE BAD SECTOR FILE FROM THE LAST TRACK, AND,
 8055                           ;       APPROVE THE USAGE OF A SECTOR BASED ON INFORMATION IN
 8056                           ;THE BAD SECTOR FILE OR ASSIGN A NEW SECTOR IF THE ONE ELECTED IS
 8057                           ;NOT AVAILABLE FOR USE.
 8058                           ;
 8059                           ;INFORMATION REQUIRED BY THE MODULE INCLUDES
 8060                           ;       .RMDCO, THE DESIRED CYLINDER,
 8061                           ;       .RMDAO, THE TRACK AND SECTOR ADDRESS,
 8062                           ;       .RMWCO, THE WORD COUNT,
 8063                           ;       .RMCS10, THE COMMAND.
 8064                           ;
 8065                           ;MODULE CALL IS AS FOLLOWS,
 8066                           ;
 8067                           ;       JSR     PC,BADSCT
 8068                           ;       BR      ???             ;RETURN HERE IF NO ERROR
 8069                           ;       TYPE    ,MESSAGE        ;RETURN HERE IF THE BAD SECTOR FILE
 8070                           ;                               CANNOT BE RECOVERED
 8071                           ;       ERROR                   THE MODULE DEFINES THE ERROR NUMBER
 8072                           ;
 8073                           ;THE MODULE IS INTENDED TO BE CALLED PRIOR TO CALLING THE BUFFER
 8074                           ;GENERATOR SUBROUTINE, AND PRESERVES THE "PUT BUFFER" SO THAT THE
 8075                           ;BUFFER NEED ONLY BE FILLED ONCE FOR THE EXECUTION OF A FORMAT
 8076                           ;OPERATION.
 8077                           ;
 8078                           ;THE MODULE RETURNS TO THE CALLING TEST WITH THE APPROVED OR ASSIGNED
 8079                           ;SECTOR IN THE PUT BUFFER AND ALSO IN LOCATIONS "ASNDA" AND "ASNDC"
 8080                           ;SO THAT A REFERENCE IS AVAILABLE TO THE TEST OUTSIDE OF THE PUT BUFFER.
 8081
 8082
 8083  034210                   BADSCT:
 8084
 8085                           ;TEST "MEDIA ENABLE" TO DETERMINE WHETHER OR NOT THE BAD SECTOR FILES
 8086                           ;HAVE BEEN RECOVERED.
 8087  034210  005737  001472           TST     MEDENB
 8088  034214  001402                   BEQ     10$
 8089  034216  000137  035434           JMP     300$            ;BAD SECTOR FILE IS AVAILABLE
 8090
 8091                           ;;*****************************************************************
 8092                           ;RECOVER THE BAD SECTOR FILE FROM THE LAST TRACK
 8093  034222                   10$:
 8094
 8095                           ;SAVE THE USER'S PUT BUFFER
 8096  034222  010046                   MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 8097  034224  005000                   CLR     R0
 8098  034226  016060  001376  102574   20$:    MOV     PUTBUF(R0),BUFFER(R0)
 8099  034234  062700  000002           ADD     #2,R0           ;ADVANCE TO NEXT BUFFER POSITION
 8100  034240  022700  000046           CMP     #46,R0          ;END OF BUFFER
 8101  034244  103370                   BHIS    20$             ;NO !!
 8102
 8103                           ;SET RETRY COUNT AND LOAD PUT BUFFER AND REGISTER INDEX TABLE
 8104  034246  012737  000003  036102   MOV     #3,500$         ;RETRY COUNT
 8105  034254  012737  002000  001404   MOV     #002000,RMDAO   ;STARTING SECTOR ADDRESS
 8106  034262  012737  001466  001432   MOV     #822.,RMDCO     ;DESIRED CYLINDER
```

```
8107  034270  012737  177376  001400          MOV     #<↑C258.+1>,RMWCO       ;WORD COUNT
8108  034276  012737  010000  001430          MOV     #FMT16,RMOFO    ;16 BIT FORMAT
8109  034304  012737  100554  001402          MOV     #MFGFIL,RMBAO   ;BUFFER ADDRESS
8110
8111  034312  012700  001533                  MOV     #PUTINX,R0      ;R0 POINTS TO REGISTER INDEX TABLE
8112  034316  112720  000006                  MOVB    #RMDA,(R0)+
8113  034322  112720  000034                  MOVB    #RMDC,(R0)+
8114  034326  112720  000002                  MOVB    #RMWC,(R0)+
8115  034332  112720  000032                  MOVB    #RMOF,(R0)+
8116  034336  112720  000004                  MOVB    #RMBA,(R0)+
8117  034342  112720  000000                  MOVB    #RMCS1,(R0)+
8118  034346  112720  000200                  MOVB    #200,(R0)+
8119  034352  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8120
8121                                  ;SET GET INDEX TABLE FOR READING STATUS
8122  034354  004737  036726                  JSR     PC,GETSTS       ;SETUP GET INDEX REGISTER FOR STATUS
8123  034360                          30$:
8124
8125                                  ;CLEAR THE DEVICE USING DRIVE CLEAR COMMAND
8126  034360  012737  000011  001376          MOV     #DRVCLR!GO,RMCS10       ;LOAD COMMAND IN PUT BUFFER
8127  034366  004737  037262                  JSR     PC,PUT          ;OUTPUT COMMAND
8128  034372  000411                          BR      45$             ;RETURN HERE IF NO ERROR
8129  034374  000401                          BR      40$             ;GET AROUND ERROR #
8130  034376  000000                  35$:    .WORD                   ;ERROR # GOES HERE
8131
8132  034400  062716  000006          40$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8133  034404  113776  034376  000000          MOVB    35$,@(SP)       ;MOVE ERROR NUMBER TO USER
8134  034412  000137  035112                  JMP     215$
8135  034416  004737  037012          45$:    JSR     PC,GET          ;GO GET STATUS
8136  034422  000411                          BR      60$             ;RETURN HERE IF NO ERROR
8137  034424  000401                          BR      55$             ;GET AROUND ERROR #
8138  034426  000000                  50$:    .WORD                   ;ERROR # GOES HERE
8139
8140  034430  062716  000006          55$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8141  034434  113776  034426  000000          MOVB    50$,@(SP)       ;MOVE ERROR # TO USERS ERROR CALL
8142  034442  000137  035112                  JMP     215$
8143
8144  034446  004737  051510          60$:    JSR     PC,DRVSTS       ;GO VERIFY DRIVE CLEAR COMMAND
8145  034452  000412                          BR      75$             ;RETURN HERE IF NO ERROR
8146  034454  000401                          BR      70$             ;GET AROUND ERROR
8147  034456  000000                  65$:    .WORD                   ;ERROR # GOES HERE
8148
8149  034460  005726                  70$:    TST     (SP)+           ;STRIP RETURN TO SUBROUTINE
8150  034462  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8151  034466  113776  034456  000000          MOVB    65$,@(SP)       ;MOVE ERROR # TO USER CALL
8152  034474  000137  035112                  JMP     215$
8153
8154  034500                          75$:
8155
8156                                  ;ISSUE A PACK ACKNOWLEDGE IF VOLUME VALID IS RESET
8157  034500  032737  000100  001340          BIT     #VV,RMDSI       ;IS VV RESET ??
8158  034506  001050                          BNE     120$            ;NO !!
8159  034510  012737  000023  001376          MOV     #PACACK!GO,RMCS10       ;LOAD COMMAND
8160  034516  004737  037262                  JSR     PC,PUT          ;GO PUT COMMAND TO DRIVE
8161  034522  000411                          BR      90$             ;RETURN HERE IF NO OUTPUT ERROR
8162  034524  000401                          BR      85$             ;GET AROUND ERROR #
```

```
8163  034526  000000                    80$:    .WORD                   ;ERROR # GOES HERE
8164
8165  034530  062716  000006            85$:    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8166  034534  113776  034526  000000            MOVB    80$,2(SP)       ;MOVE ERROR # TO ERROR CALL
8167  034542  000137  035112                    JMP     215$
8168  034546  004737  037012            90$:    JSR     PC,GET          ;GO GET STATUS FOR PACK ACK
8169  034552  000411                            BR      105$            ;RETURN HERE IF NO ERROR
8170  034554  000401                            BR      100$            ;GET AROUND ERROR #
8171  034556  000000                    95$:    .WORD                   ;ERROR # GOES HERE
8172
8173  034560  062716  000006            100$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8174  034564  113776  034556  000000            MOVB    95$,2(SP)       ;MOVE ERROR # TO CALL
8175  034572  000137  035112                    JMP     215$
8176
8177  034576  004737  047152            105$:   JSR     PC,ACKSTS       ;GO VERIFY ACKNOWLEDGE STATUS
8178  034602  000412                            BR      120$            ;RETURN HERE IF NO ERROR
8179  034604  000401                            BR      115$            ;GET AROUND ERROR #
8180  034606  000000                    110$:   .WORD                   ;ERROR # GOES HERE
8181
8182  034610  005726                    115$:   TST     (SP)+           ;STRIP RETURN TO SUBROUTINE
8183  034612  062716  000006                    ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8184  034616  113776  034606  000000            MOVB    110$,2(SP)      ;MOVE ERROR # TO USERS ERROR CALL
8185  034624  000137  035112                    JMP     215$
8186
8187  034630                            120$:
8188
8189                                    ;RECALIBRATE THE DRIVE IF PIP IS SET
8190  034630  032737  020000  001340            BIT     #PIP,RMDSI      ;IS PIP SET ??
8191  034636  001452                            BEQ     165$            ;NO !!
8192
8193  034640  012737  000007  001376            MOV     #RECAL!GO,RMCS10        ;LOAD RECALIBRATE COMMAND
8194  034646  004737  037262                    JSR     PC,PUT          ;PUT THE RECAL COMMAND
8195  034652  000411                            BR      135$            ;RETURN HERE IF NO ERROR
8196  034654  000401                            BR      130$            ;GET AROUND ERROR #
8197  034656  000000                    125$:   .WORD                   ;ERROR # GOES HERE
8198
8199  034660  062716  000006            130$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8200  034664  113776  034656  000000            MOVB    125$,2(SP)      ;MOVE ERROR # TO USERS CALL
8201  034672  000137  035112                    JMP     215$
8202  034676                            135$:
8203  034676  004737  037622                    JSR     PC,TIMOUT       ;WAIT FOR RECALIBRATE TO COMPLETE
8204  034702  004737  037012                    JSR     PC,GET          ;GO GET RECAL STATUS
8205  034706  000411                            BR      150$            ;RETURN HERE IF NO ERROR
8206  034710  000401                            BR      145$            ;GET AROUND ERROR #
8207  034712  000000                    140$:   .WORD                   ;ERROR # GOES HERE
8208
8209  034714  062716  000006            145$:   ADD     #6,(SP)         ;MOVE SP TO USERS ERROR CALL
8210  034720  113776  034712  000000            MOVB    140$,2(SP)      ;MOVE ERROR TO USERS CALL
8211  034726  000137  035112                    JMP     215$
8212
8213  034732  004737  047746            150$:   JSR     PC,RCLSTS       ;GO VERIFY RECALIBRATE STATUS
8214  034736  000412                            BR      165$            ;RETURN HERE IF NO ERROR
8215  034740  000401                            BR      160$            ;GET AROUND ERROR #
8216  034742  000000                    155$:   .WORD                   ;ERROR # GOES HERE
8217
8218  034744  005726                    160$:   TST     (SP)+           ;STRIP RETURN TO SUBROUTINE
```

```
 8219  034746  062716  000006              ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
 8220  034752  113776  034742  000000      MOVB    155$,@(SP)       ;MOVE ERROR # TO USERS CALL
 8221  034760  000137  035112              JMP     215$
 8222
 8223  034764                          165$:
 8224
 8225                                  ;READ THE SECTOR IDENTIFIED BY RMDAO, INCLUDING HEADER AND DATA
 8226
 8227  034764  012737  000073  001376      MOV     #RH!GO,RMCS1O    ;LOAD READ HEADER AND DATA COMMAND
 8228  034772  004737  037262              JSR     PC,PUT           ;PUT COMMAND
 8229  034776  000411                      BR      180$             ;RETURN HERE IF NO ERROR
 8230  035000  000401                      BR      175$             ;GET AROUND ERROR #
 8231  035002  000000                  170$:   .WORD                    ;ERROR # GOES HERE
 8232
 8233  035004  062716  000006          175$:   ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
 8234  035010  113776  035002  000000      MOVB    170$,@(SP)       ;MOVE ERROR # TO USERS ERROR CALL
 8235  035016  000137  035112              JMP     215$
 8236
 8237  035022  004737  037622          180$:   JSR     PC,TIMOUT        ;WAIT FOR READ OPERATION TO COMPLETE
 8238  035026  004737  037012              JSR     PC,GET           ;GO GET STATUS FOR READ OPERATION
 8239  035032  000411                      BR      195$             ;RETURN HERE IF NO ERROR
 8240  035034  000401                      BR      190$             ;GET AROUND ERROR #
 8241  035036  000000                  185$:   .WORD                    ;ERROR # GOES HERE
 8242
 8243  035040  062716  000006          190$:   ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
 8244  035044  113776  035036  000000      MOVB    185$,@(SP)       ;MOVE ERROR # TO CALL
 8245  035052  000137  035112              JMP     215$
 8246
 8247  035056  004737  052312          195$:   JSR     PC,DTASTS        ;GO VERIFY RESULTS OF READ OPERATION
 8248  035062  000412                      BR      210$             ;RETURN HERE IF NO ERROR
 8249  035064  000401                      BR      205$             ;GET AROUND ERROR #
 8250  035066  000000                  200$:   .WORD                    ;ERROR # GOES HERE
 8251
 8252  035070  005726                  205$:   TST     (SP)+            ;STRIP RETURN ADDRESS TO SUBROUTINE
 8253  035072  062716  000006              ADD     #6,(SP)          ;MOVE SP TO USERS ERROR CALL
 8254  035076  113776  035066  000000      MOVB    200$,@(SP)       ;MOVE ERROR # TO USERS CALL
 8255  035104  000137  035112              JMP     215$
 8256
 8257  035110  000456                  210$:   BR      240$
 8258
 8259  035112                          215$:
 8260
 8261                                  ;AN ERROR HAS BEEN DETECTED IN TRYING TO READ THE BAD SECTOR FILE.
 8262                                  ;THE SECTOR WILL BE RETRIED IF POSSIBLE.
 8263
 8264  035112  032737  010000  001340      BIT     #MOL,RMDSI       ;IS MEDIUM ON LINE ??
 8265  035120  001446                      BEQ     230$             ;YES !!
 8266
 8267  035122  005337  036102              DEC     500$             ;DECREMENT RETRY COUNT
 8268  035126  100037                      BPL     225$             ;RETRY IF COUNT NOT NEG
 8269
 8270                                  ;THE RETRY COUNT HAS EXPIRED - SEE IF THE ERROR IS MEDIA RELATED
 8271  035130  013746  001342              MOV     RMER1I,-(SP)     ;GET ER1
 8272  035134  042716  100720              BIC     #DCK!HCRC!HCE!FER!ECH,(SP)
 8273  035140  032737  000010  001370      BIT     #DPE,RMER2I      ;WAS THER A DATA PARITY ERROR ??
 8274  035146  001402                      BEQ     220$             ;NO !!
```

```
8275  035150  042716  000010            BIC     #PAR,(SP)          ;YES - CLEAR PAR
8276  035154  005726                220$: TST     (SP)+              ;ARE THERE ANY ERRORS NOT DUE TO MEDIA ?
8277  035156  001027                      BNE     230$               ;YES !!
8278  035160  013746  001370            MOV     RMER2I,-(SP)       ;GET ER2
8279  035164  042726  100010            BIC     #BSE!DPE,(SP)+     ;CLEAR MEDIA ERRORS
8280  035170  001022                      BNE     230$               ;BRANCH IF NONMEDIA ERRORS DETECTED
8281
8282                                    ;THE ERRORS DETECTED WHILE TRYING TO RECOVER THE BAD SECTOR FILE ARE
8283                                    ;DUE TO THE MEDIA.  SEE IF THE BAD SECTOR FILE CAN BE RECOVERED FROM
8284                                    ;ANOTHER AREA ON THE LAST TRACK
8285
8286  035172  062737  000002  001404      ADD     #2,RMDAO           ;ADVANCE SECTOR ADDRESS
8287  035200  122737  000012  001404      CMPB    #10.,RMDAO         ;QUIT IF ALL MFG SECTORS HAVE BEEN'
8288  035206  001413                      BEQ     230$               ;TRIED
8289  035210  122737  000040  001404      CMPB    #32.,RMDAO         ;QUIT IF ALL USER SECTORS HAVE
8290  035216  001407                      BEQ     230$               ;BEEN TRIED
8291  035220  012737  000003  036102      MOV     #3,500$            ;REINSTATE RETRY COUNT FOR THIS SECTOR
8292  035226  162716  000006        225$: SUB     #6,(SP)            ;MOVE SP BACK TO NO ERROR
8293  035232  000137  034360            JMP     30$                ;RETRY THE READ OPERATION
8294
8295  035236                          230$:
8296
8297                                    ;THE BAD SECTOR FILE CANNOT BE READ
8298  035236  162716  000004            SUB     #4,(SP)            ;MOVE SP TO ERROR RETURN
8299  035242  000137  036076            JMP     410$               ;GO TO MODULE EXIT
8300
8301  035246                          240$:
8302
8303                                    ;THE SECTOR WAS RECOVERED WITHOUT ERROR -
8304                                    ;       READ THE USER FILE IF THIS IS THE MFG FILE
8305                                    ;       OR ELSE DONE
8306
8307  035246  122737  000011  001404      CMPB    #9.,RMDAO          ;WAS THE USER FILE READ ??
8308  035254  103451                      BLO     260$               ;YES - READ IS COMPLETE
8309
8310                                    ;IF 144 IS IMPLEMENTED THEN READ THE USER FILE -
8311                                    ;       ELSE DUMMY THE BAD SECTOR FILE
8312  035256  032737  140000  100554      BIT     #MSE!USE,MFGFIL    ;ARE THE BAD SECTOR FLAGS OFF ??
8313  035264  001410                      BEQ     250$               ;YES - 144 IS DISABLED
8314
8315  035266  012737  002012  001404      MOV     #002012,RMDAO      ;READ THE USER FILE
8316  035274  012737  000003  036102      MOV     #3,500$            ;RELOAD THE RETRY COUNT FOR THIS SECTOR
8317  035302  000137  034360            JMP     30$                ;GO READ THE USER FILE
8318  035306                          250$:
8319
8320                                    ;DUMMY THE BAD SECTOR FILES
8321  035306  010046                    MOV     R0,-(SP)           ;;PUSH R0 ON STACK
8322  035310  010146                    MOV     R1,-(SP)           ;;PUSH R1 ON STACK
8323  035312  012701  000374            MOV     #252.,R1           ;R1 = NUMBER OF ENTRIES IN FILES
8324  035316  012700  000014            MOV     #14,R0             ;R0 = ADDRESS INDEX TO FILE  STORAGE
8325  035322  012760  177777  100554  255$: MOV     #-1,MFGFIL(R0)     ;ENTER ALL ONES IN MFG FILE
8326  035330  012760  177777  101564      MOV     #-1,USRFIL(R0)     ;ENTER ALL ONES IN USER FILE
8327  035336  062700  000002            ADD     #2,R0              ;ADVANCE ADDRESS
8328  035342  005301                    DEC     R1                 ;DECREMENT COUNT
8329  035344  001366                    BNE     255$               ;CONTINUE IF NOT DONE
8330  035346  012701  000006            MOV     #6.,R1             ;CLEAR ID, SERIAL NUMBER ETC
```

```
8331  035352  005000                          CLR     R0
8332  035354  005060  100554          256$:   CLR     MFGFIL(R0)
8333  035360  005060  101564                  CLR     USRFIL(R0)
8334  035364  062700  000002                  ADD     #2,R0
8335  035370  005301                          DEC     R1
8336  035372  001370                          BNE     256$
8337
8338  035374  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
8339  035376  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8340  035400                          260$:
8341
8342                                  ;SET MEDIA ENABLE AND RESTORE THE USERS PUT BUFFER
8343  035400  012737  177777  001472          MOV     #-1,MEDENB
8344
8345  035406  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8346  035410  005000                          CLR     R0              ;R0 IS REGISTER INDEX
8347  035412  016060  102574  001376  265$:   MOV     BUFFER(R0),PUTBUF(R0)
8348  035420  062700  000002                  ADD     #2,R0           ;ADVANCE R0
8349  035424  022700  000046                  CMP     #46,R0          ;DONE ??
8350  035430  103370                          BHIS    265$
8351
8352  035432  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8353  035434                          270$:
8354
8355  035434                          300$:
8356
8357                                  ;;*****************************************************************
8358                                  ;VERIFY THAT THE DESIRED SECTOR IS NOT IN THE MFG BAD SECTOR FILE
8359                                  ;AND NOT IN THE USERS BAD SECTOR FILE.  ASSIGN A NEW SECTOR IF THE
8360                                  ;SECTOR IS IN EITHER FILE.
8361
8362                                  ;LOAD INITIAL VARIABLES AND COMPUTE THE NUMBER OF SECTORS
8363  035434  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8364  035436  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
8365  035440  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8366  035442  013737  001432  001474          MOV     RMDCO,ASNDC     ;LOAD REQUESTED CYLINDER, TRACK,
8367  035450  013737  001404  001476          MOV     RMDAO,ASNDA     ;AND SECTOR ADDRESS IN ASSIGNED STORAGE
8368  035456  005002                          CLR     R2              ;R2 = NUMBER OF SECTORS
8369  035460  013700  001400                  MOV     RMWCO,R0        ;R0 = WORD COUNT
8370  035464  005300                          DEC     R0              ;CONVERT FROM 2'S COMPLEMENT
8371  035466  005100                          COM     R0
8372  035470  012701  000400                  MOV     #256.,R1        ;R1 = NUMBER OF WORDS PER SECTOR
8373  035474  032737  000002  001376          BIT     #BIT1,RMCS10    ;IS THIS A HEADER AND DATA COMMAND ??
8374  035502  001402                          BEQ     305$            ;NO !!
8375  035504  012701  000402                  MOV     #258.,R1        ;CHANGE WORDS PER SECTOR
8376  035510  020100                  305$:   CMP     R1,R0           ;IS THERE A FULL SECTOR ??
8377  035512  101404                          BLOS    310$            ;YES !!
8378  035514  005700                          TST     R0              ;IS R0 ZERO ??
8379  035516  001405                          BEQ     315$            ;YES !!
8380  035520  005202                          INC     R2              ;INCREMENT FOR PARTIAL SECTOR
8381  035522  000403                          BR      315$
8382  035524  160100                  310$:   SUB     R1,R0           ;SUBTRACT ONE SECTOR FROM WORD COUNT
8383  035526  005202                          INC     R2              ;INCREMENT SECTOR COUNT
8384  035530  000767                          BR      305$
8385  035532  010237  036102          315$:   MOV     R2,500$         ;SAVE SECTOR COUNT
8386
```

```
8387  035536                                316$:
8388
8389                                        ;LOAD PARAMETERS AND SEARCH THE MFG AND USER BAD SECTOR FILES FOR
8390                                        ;THE ASSIGNED SECTOR AND ADJACENT SECTORS IF THE SECTOR COUNT IS
8391                                        ;MORE THAN ONE.
8392
8393  035536  012737  100570  036112        MOV     #MFGFIL+14,540$ ;MOVE THE STARTING ADDRESS OF MFG
8394                                                                ;FILE TO BASE ADDRESS STORAGE
8395  035544  013737  001474  036106  320$:  MOV     ASNDC,520$      ;LOAD COMPARING CYLINDER ADDRESS
8396  035552  013737  001476  036110        MOV     ASNDA,530$      ;LOAD COMPARING TRACK, SECTOR ADDRESS
8397  035560  013737  036102  036104        MOV     500$,510$       ;LOAD NUMBER OF SECTORS TO CONFIRM
8398
8399                                        ;SETUP FOR A BINARY SEARCH OF THE CURRENT FILE FOR THE COMPARING
8400                                        ;CYLINDER, TRACK AND SECTOR ADDRESS
8401  035566  013700  036112        325$:  MOV     540$,R0         ;LOAD THE BASE ADDRESS IN R0
8402  035572  012701  000376               MOV     #<<127.*4>/2>,R1        ;R1 = LENGTH OF FILE
8403  035576  060100                        ADD     R1,R0           ;START BINARY DIVIDE AT CENTER
8404  035600  021037  036106        330$:  CMP     (R0),520$       ;DOES TABLE ENTRY = COMPARING CYLINDER ?
8405  035604  001405                        BEQ     345$            ;YES !!
8406  035606  101002                        BHI     340$            ;BRANCH IF ENTRY > COMPARING CYLINDER
8407  035610                        335$:
8408
8409                                        ;FILE ENTRY IS LESS THAN COMPARING CYLINDER (OR TRACK AND SECTOR),
8410                                        ;SO ADD DISPLACEMENT TO CURRENT POSITION
8411  035610  060100                        ADD     R1,R0
8412  035612  000410                        BR      350$
8413  035614                        340$:
8414
8415                                        ;FILE ENTRY IS GREATER THAN COMPARING CYLINDER SO SUBTRACT DISPLACEMENT
8416                                        ;FROM CURRENT POSITION
8417  035614  160100                        SUB     R1,R0
8418  035616  000406                        BR      350$
8419  035620                        345$:
8420
8421                                        ;FILE ENTRY EQUALS COMPARING CYLINDER.  SEE IF THE NEXT ENTRY EQUALS
8422                                        ;THE COMPARING TRACK, AND SECTOR.
8423  035620  026037  000002  036110        CMP     2(R0),530$      ;ARE THEY EQUAL ??
8424  035626  001413                        BEQ     360$            ;YES !!
8425  035630  101371                        BHI     340$            ;BRANCH IF HIGHER
8426  035632  000766                        BR      335$            ;BRANCH IF LOWER
8427  035634                        350$:
8428
8429                                        ;THE POINTER (R0) HAS BEEN ADJUSTED.  HALVE THE DISPLACEMENT AND
8430                                        ;CONTINUE THE SEARCH IF DISPLACEMENT NOT ZERO
8431  035634  005701                        TST     R1              ;IS DISPLACEMENT ZERO ??
8432  035636  001440                        BEQ     370$            ;YES !!
8433  035640  006201                        ASR     R1              ;HALVE THE DISLPACEMENT
8434  035642  042701  000003               BIC     #3,R1
8435  035646  020037  036112               CMP     R0,540$         ;IS THE NEW POINTER WITHIN BOUNDS ??
8436  035652  103432                        BLO     370$            ;NO !!
8437  035654  000751                        BR      330$            ;CONTINUE SEARCH
8438  035656                        360$:
8439
8440                                        ;THE COMPARING CYLINDER, TRACK AND SECTOR IS IN THE BAD SECTOR FILE.
8441                                        ;ADVANCE THE ASSIGNED SECTOR AND START THE SEARCH ALL OVER.
8442  035656  105237  001476               INCB    ASNDA           ;INCREMENT SECTOR
```

```
8443  035662  122737  000037  001476          CMPB    #31.,ASNDA      ;SECTOR OK ??
8444  035670  103022                          BHIS    365$            ;YES !!
8445  035672  105037  001476                  CLRB    ASNDA           ;CLEAR SECTOR AND ADVANCE TRACK
8446  035676  105237  001477                  INCB    ASNDA+1
8447  035702  122737  000004  001477          CMPB    #4,ASNDA+1      ;TRACK OK ??
8448  035710  103012                          BHIS    365$            ;YES !!
8449  035712  005037  001476                  CLR     ASNDA           ;CLEAR TRACK AND SECTOR
8450  035716  005237  001474                  INC     ASNDC           ;INCREMENT CYLINDER
8451  035722  022737  001466  001474          CMP     #822.,ASNDC     ;CYLINDER OK ??
8452  035730  103002                          BHIS    365$            ;YES !!
8453  035732  005037  001474                  CLR     ASNDC
8454  035736  000677                  365$:   BR      316$            ;REPEAT SEARCH
8455
8456  035740                          370$:
8457
8458                                  ;THE COMPARING SECTOR IS NOT IN THE BAD SECTOR FILES.  DECREMENT THE
8459                                  ;NUMBER OF SECTORS TO COMPARE AND SEARCH THE NEXT SECTOR IF THE NUMBER
8460                                  ;IS NOT ZERO.
8461
8462  035740  005337  036104                  DEC     510$            ;DECREMENT NUMBER OF SECTORS TO COMPARE
8463  035744  001432                          BEQ     380$            ;DONE IF ZERO
8464  035746  105237  036110                  INCB    530$            ;INCREMENT THE COMPARING SECTOR
8465  035752  122737  000037  036110          CMPB    #31.,530$       ;SECTOR OK ??
8466  035760  103022                          BHIS    375$            ;YES !!
8467  035762  105037  036110                  CLRB    530$            ;CLEAR SECTOR
8468  035766  105237  036111                  INCB    530$+1          ;INCREMENT TRACK
8469  035772  122737  000004  036111          CMPB    #4,530$+1       ;TRACK OK ??
8470  036000  103012                          BHIS    375$            ;YES !!
8471  036002  005037  036110                  CLR     530$            ;CLEAR SECTOR TRACK
8472  036006  005237  036106                  INC     520$            ;INCREMENT CYLINDER
8473  036012  022737  001466  036106          CMP     #822.,520$      ;CYLINDER OK ??
8474  036020  103002                          BHIS    375$            ;YES !!
8475  036022  005037  036106                  CLR     520$            ;CLEAR CYLINDER
8476  036026  000137  035566          375$:   JMP     325$            ;SEARCH NEXT SECTOR
8477
8478  036032                          380$:
8479
8480                                  ;THE ASSIGNED SECTOR (AND REQUIRED ADJACENT SECTORS) ARE NOT IN
8481                                  ;THE BAD SECTOR FILE JUST SEARCHED.  SEARCH THE USER FILE IF IT
8482                                  ;HAS NOT YET BEEN SEARCHED, ELSE ASSIGN THE SECTOR AND RETURN TO USER.
8483
8484  036032  022737  101600  036112          CMP     #USRFIL+14,540$ ;TEST BASE ADDRESS
8485  036040  001405                          BEQ     400$            ;DONE IF BASE = USER
8486  036042  012737  101600  036112          MOV     #USRFIL+14,540$ ;LOAD BASE ADDRESS
8487  036050  000137  035544          390$:   JMP     320$            ;SEARCH USER FILE
8488
8489
8490  036054                          400$:
8491
8492                                  ;ASSIGN THE SECTOR AND RETURN TO USER
8493  036054  013737  001474  001432          MOV     ASNDC,RMDC0     ;LOAD CYLINDER
8494  036062  013737  001476  001404          MOV     ASNDA,RMDA0     ;LOAD TRACK AND SECTOR
8495  036070  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
8496  036072  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
8497  036074  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8498  036076  000240                  410$:   NOP
```

```
8499  036100  000207                        RTS     PC
8500
8501                                        ;THE FOLLOWING ARE STORAGE LOCATIONS FOR THE MODULE
8502
8503  036102  000000                        500$:   .WORD   ;RETRY COUNT/ NUMBER OF SECTORS REQUIRED
8504  036104  000000                        510$:   .WORD   ;NUMBER OF SECTORS TO COMPARE
8505  036106  000000                        520$:   .WORD   ;COMPARING CYLINDER
8506  036110  000000                        530$:   .WORD   ;COMPARING TRACK AND SECTOR
8507  036112  000000                        540$:   .WORD   ;BASE ADDRESS OF BAD SECTOR FILE BEING SEARCHED
```

```
 8508                                   .SBTTL  BUFFER GENERATOR SUBROUTINE
 8509
 8510                                   ; THIS SUBROUTINE GENERATES A DATA BUFFER FOR WRITE COMMANDS.  THE
 8511                                   ; BUFFER STARTS AT RMBA AND IS RMWC WORDS LONG.  THE BUFFER
 8512                                   ; CONTAINS A REPETITIVE DATA PATTERN CONSISTING OF $TMP1 WORDS
 8513                                   ; FROM THE DATA PATTERN TABLE, BEGINNING AT ADDRESS $TMP0.
 8514                                   ; HEADER INFORMATION FOR THE BUFFER IS EXTRACTED FROM RMDC,
 8515                                   ; RMDA AND RMOF.
 8516
 8517                                   ;CALL:
 8518                                   ;(1)    JSR     PC,GENBUF
 8519                                   ;(2)    ??                              RETURN HERE
 8520
 8521  036114                           GENBUF:
 8522  036114  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 8523  036116  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 8524  036120  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 8525  036122  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 8526  036124  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 8527  036126  013700  001402                  MOV     RMBA0,R0        ;LOAD DATA BUFFER ADDRESS
 8528  036132  013701  001400                  MOV     RMWC0,R1        ;LOAD WORD COUNT
 8529  036136  013737  001432  036354          MOV     RMDC0,60$       ;LOAD STARTING CYLINDER ADDRESS
 8530  036144  013737  001404  036356          MOV     RMDA0,65$       ;LOAD STARTING TRACK,SECTOR ADDRESS
 8531  036152  032737  000002  001376  10$:    BIT     #BIT1,RMCS1     ;WRITE HEADER & DATA??
 8532  036160  001450                          BEQ     25$             ;NO!!
 8533  036162  013710  036354                  MOV     60$,(R0)        ;WRITE HEADER WORD #1
 8534  036166  042710  176000                  BIC     #^C<CYLMSK>,(R0)
 8535  036172  052710  140000                  BIS     #MSE!USE,(R0)   ;SET (DISABLE) BAD SECTOR FLAGS
 8536  036176  012702  000035                  MOV     #29.,R2         ;R2 = MAXIMUM SECTOR ADDRESS
 8537  036202  032737  010000  001430          BIT     #FMT16,RMOF0    ;16 BIT FORMAT??
 8538  036210  001404                          BEQ     15$             ;NO!!
 8539  036212  052710  010000                  BIS     #FMT16,(R0)     ;SET FORMAT BIT IN HEADER
 8540  036216  012702  000037                  MOV     #31.,R2         ;CHANGE MAXIMUM SECTOR ADDRESS
 8541  036222  005201                  15$:    INC     R1              ;INCREMENT WORD COUNT
 8542  036224  001444                          BEQ     50$             ;EXIT IF DONE
 8543  036226  062700  000002                  ADD     #2,R0           ;MOVE R0 TO HEADER WORD #2
 8544  036232  013720  036356                  MOV     65$,(R0)+       ;WRITE HEADER WORD #2
 8545  036236  005201                          INC     R1              ;INCREMENT WORD COUNT AND
 8546  036240  001436                          BEQ     50$             ;EXIT IF DONE
 8547  036242  012703  036356                  MOV     #65$,R3         ;ADVANCE SECTOR ADDRESS
 8548  036246  105213                          INCB    (R3)
 8549  036250  120213                          CMPB    R2,(R3)         ;SECTOR OVERFLOW ??
 8550  036252  103013                          BHIS    25$             ;NO !!
 8551  036254  105013                          CLRB    (R3)            ;YES - CLEAR SECTOR ADDRESS
 8552  036256  105263  000001                  INCB    1(R3)           ;ADVANCE TRACK ADDRESS
 8553  036262  122763  000004  000001          CMPB    #4,1(R3)        ;TRACK OVERFLOW ??
 8554  036270  103004                          BHIS    25$             ;NO !!
 8555  036272  105063  000001                  CLRB    1(R3)           ;YES - CLEAR TRACK ADDRESS
 8556  036276  105237  036354                  INCB    60$             ;ADVANCE CYLINDER ADDRESS
 8557  036302  012704  000400          25$:    MOV     #256.,R4                ;LOAD SECTOR DATA COUNT
 8558  036306  013702  001174          30$:    MOV     $TMP0,R2        ;LOAD PATTERN ADDRESS
 8559  036312  013703  001176                  MOV     $TMP1,R3        ;LOAD PATTERN COUNT
 8560  036316  012220                  40$:    MOV     (R2)+,(R0)+     ;WRITE DATA PATTERN
 8561  036320  005201                          INC     R1              ;INCREMENT WORD COUNT AND
 8562  036322  001405                          BEQ     50$             ;EXIT IF DONE
 8563  036324  005304                          DEC     R4              ;DECREMENT SECTOR COUNT
```

```
 8564  036326  001711                BEQ     10$             ;START NEXT SECTOR IF 0
 8565  036330  005303                DEC     R3              ;DECREMENT PATTERN COUNT
 8566  036332  001765                BEQ     30$             ;RESTART PATTERN IF 0
 8567  036334  000770                BR      40$             ;CONTINUE DATA PATTERN
 8568
 8569  036336                    50$:
 8570  036336  012604                MOV     (SP)+,R4        ;;POP STACK INTO R4
 8571  036340  012603                MOV     (SP)+,R3        ;;POP STACK INTO R3
 8572  036342  012602                MOV     (SP)+,R2        ;;POP STACK INTO R2
 8573  036344  012601                MOV     (SP)+,R1        ;;POP STACK INTO R1
 8574  036346  012600                MOV     (SP)+,R0        ;;POP STACK INTO R0
 8575  036350  000240                NOP
 8576  036352  000207                RTS     PC
 8577
 8578  036354  000000            60$:    .WORD               ;CYLINDER ADDRESS STORAGE
 8579  036356  000000            65$:    .WORD               ;TRACK, SECTOR ADDRESS STORAGE
```

```
                        .SBTTL  COMPARE BUFFER SUBROUTINE

                        ;THIS SUBROUTINE COMPARES THE CONTENTS OF BUFONE AND BUFTWO,
                        ;ASSUMING THAT BUFONE IS THE BUFFER FROM WHICH DATA WAS WRITTEN
                        ;AND BUFTWO IS THE BUFFER TO WHICH DATA WAS READ.  ERRORS IN BUFFER
                        ; COMPARISON ARE REPORTED TO THE USER VIA THE USER'S ERROR CALL.

                        ;CALL:
                        ;(1)    JSR     PC,CMPBUF
                        ;(2)    .WORD   WRITE BUFFER ADDRESS
                        ;       .WORD   READ BUFFER ADDRESS
                        ;(3)    BR      ??              RETURN HERE IF NO ERROR
                        ;(4)    NOP                     RETURN HERE IF ERROR
                        ;(5)    ERROR                   ERROR DEFINED BY SUBROUTINE
                        ;(6)    ???

036360                  CMPBUF:
036360  010046                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
036362  010146                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
036364  010246                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
036366  010346                 MOV     R3,-(SP)        ;;PUSH R3 ON STACK

036370  005037  036724         CLR     150$            ;CLEAR CORRECTION FLAG
                        ;DETERMINE IF DATA SHOULD BE CORRECTED
036374  032737  100000  001342 BIT     #DCK,RMER1I     ;WAS THERE A DATA CHECK ??
036402  001465                 BEQ     80$             ;NO !!
036404  032737  000100  001342 BIT     #ECH,RMER1I     ;IS IT A HARD ERROR ??
036412  001061                 BNE     80$             ;YES !!
036414  033737  004000  001360 BIT     ECI,RMOFI       ;WAS ECC CORRECTION ALLOWED ??
036422  001055                 BNE     80$             ;NO !!
036424  032737  010000  001360 BIT     #FMT16,RMOFI    ;IS THIS 16 BIT FORMAT ??
036432  001451                 BEQ     80$             ;NO !!

                        ;CORRECT DCTA USING ECC INFORMATION
036434  013700  001402         MOV     RMBAO,R0        ;R0 = STARTING BUFFER ADDRESS
036440  013701  001372         MOV     RMEC1I,R1       ;R1 = ECC POSITION
036444  052737  100000  036724 BIS     #BIT15,150$     ;SET CORRECTION FLAG
                        ;MOVE R0 TO WORD BOUNDARY OF ERROR BURST

036452  022701  000020         10$:    CMP     #16.,R1         ;IS BIT POSITION > 1 WORD
036456  103005                 BHIS    20$             ;NO !!
036460  162701  000020         SUB     #16.,R1         ;SUBTRACT 1 WORDS WORTH
036464  062700  000002         ADD     #2,R0           ;ADVANCE BUFFER ADDRESS 1 WORD
036470  000770                 BR      10$
036472  012702  000001         20$:    MOV     #1,R2           ;R2 = BIT POINTER
036476  010203                 MOV     R2,R3           ;R3 = BIT NUMBER
                        ;MOVE R2 TO STARTING BIT OF ERROR BURST

036500  020301                 30$:    CMP     R3,R1           ;IS R3 SAME AS R1 ??
036502  001403                 BEQ     35$             ;YES !!
036504  006302                 ASL     R2              ;SHIFT BIT POINTER
036506  005203                 INC     R3              ;INCREMENT BIT NUMBER
036510  000773                 BR      30$
036512  012703  000013         35$:    MOV     #11.,R3         ;R3 = LENGTH OF ERROR BURST

                        ;CORRECT THE ERROR BURST
```

```
8636  036516  030237  001374          40$:    BIT     R2,RMEC2I       ;IS THIS BIT SET IN ECC PATTERN ??
8637  036522  001405                          BEQ     60$             ;NO - DO NOT CORRECT THIS BIT
8638  036524  030210                          BIT     R2,(R0)         ;IS THE BIT PRESENTLY SET ??
8639  036526  001402                          BEQ     50$             ;NO
8640  036530  040210                          BIC     R2,(R0)         ;RESET THE BIT
8641  036532  000401                          BR      60$
8642  036534  050210                  50$:    BIS     R2,(R0)         ;SET THE BIT
8643  036536  006302                  60$:    ASL     R2              ;SHIFT TO NEXT BIT
8644  036540  001004                          BNE     70$
8645  036542  012702  000001                  MOV     #1,R2           ;CONTINUE WITH FIRST BIT OF NEXT WORD
8646  036546  062700  000002                  ADD     #2,R0
8647  036552  005303                  70$:    DEC     R3              ;END OF BURST ??
8648  036554  001360                          BNE     40$             ;NO !!
8649  036556  017600  000010          80$:    MOV     @10(SP),R0              ;R0 = WRITE BUFFER
8650  036562  062766  000002  000010          ADD     #2,10(SP)               ;MOVE SP TO READ ADDRESS
8651  036570  017601  000010                  MOV     @10(SP),R1              ;R1 = READ BUFFER
8652  036574  062766  000002  000010          ADD     #2,10(SP)               ;MOVE SP TO RETURN ADDRESS
8653  036602  013702  001330                  MOV     RMWCI,R2                ;R2 = NUMBER OF WORDS TRANSFER
8654  036606  163702  001400                  SUB     RMWCO,R2
8655  036612  022021                  90$:    CMP     (R0)+,(R1)+             ;COMPARE DATA WORDS
8656  036614  001003                          BNE     100$                    ;EXIT IF NOT EQUAL
8657  036616  005302                          DEC     R2                      ;DECREMENT WORD COUNT
8658  036620  001374                          BNE     90$                     ;CONTINUE IF NOT DONE
8659  036622  000433                          BR      110$                    ;DONE COMPARE - NO ERROR
8660  036624                          100$:
8661  036624  014037  001140                  MOV     -(R0),$GDDAT            ;STORE GOOD DATA FOR TYPEOUT
8662  036630  014137  001142                  MOV     -(R1),$BDDAT            ;STORE BAD DATA FOR TYPEOUT
8663  036634  010037  001134                  MOV     R0,$GDADR               ;STORE ADDRESS OF GOOD DATA
8664  036640  010137  001136                  MOV     R1,$BDADR               ;STORE ADDRESS OF BAD DATA
8665  036644  010237  001174                  MOV     R2,$TMP0                ;STORE WORD COUNT OF ERROR
8666  036650  062766  000004  000010          ADD     #4,10(SP)               ;MOVE SP TO USER'S ERROR CALL
8667  036656  112776  000336  000010          MOVB    #336,@10(SP)            ;WRITE ERROR NUMBER IN CALL
8668
8669                          ;CHANGE ERROR NUMBER IF ECC CORRECTION FAILED
8670  036664  032737  100000  036724          BIT     #BIT15,150$     ;WAS ECC CORRECTION USED ??
8671  036672  001403                          BEQ     105$            ;NO !!
8672  036674  112776  000163  000010          MOVB    #163,@10(SP)    ;ECC CORRECTION FAILED
8673  036702  162766  000002  000010  105$:   SUB     #2,10(SP)               ;MOVE SP TO RETURN IF ERROR
8674  036710  000240                          NOP
8675  036712                          110$:
8676  036712  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
8677  036714  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
8678  036716  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
8679  036720  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8680  036722  000207                          RTS     PC                      ;RETURN TO USER
8681
8682  036724  000000                  150$:   .WORD                   ;ECC CORRECTION FLAG
```

```
                                        .SBTTL  GET STATUS SUBROUTINE
                                        ;
                                        ;THIS SUBROUTINE SETS UP THE "GET INDEX TABLE" AND THE "GET
                                        ;BUFFER" FOR READING ALL SUBSYSTEM REGISTERS VIA THE GET SUBROUTINE
                                        ;AND THEN RETURNS TO THE USER.
                                        ;
                                        ;CALL:  JSR     PC,GETSTS
                                        ;       ???                     RETURN HERE
                                        ;
036726                                  GETSTS:
036726  010046                                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
036730  010146                                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
036732  010246                                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
036734  012700  001504                         MOV     #GETINX,R0      ;R0= ADDRESS OF INDEX TABLE
036740  012701  001376                         MOV     #RMEC2I+2,R1    ;R1 = ADDRESS OF GET BUFFER
036744  012702  000046                         MOV     #RMEC2,R2       ;R2 = REGISTER INDE
036750  110220                          2$:    MOVB    R2,(R0)+        ;WRITE REGISTER INDEX IN TABLE
036752  005041                                 CLR     -(R1)           ;CLEAR CORRESPONDING LOCATION
036754  162702  000002                  3$:    SUB     #2,R2           ;DECREMENT TO NEXT INDEX
036760  100405                                 BMI     4$              ;BRANCH OUT IF DONE
036762  022702  000022                         CMP     #RMDB,R2        ;DONT WRITE RMDB INDEX
036766  001370                                 BNE     2$
036770  005041                                 CLR     -(R1)
036772  000770                                 BR      3$
036774  112720  000200                  4$:    MOVB    #200,(R0)+      ;WRITE TERMINATOR
037000  012602                                 MOV     (SP)+,R2        ;;POP STACK INTO R2
037002  012601                                 MOV     (SP)+,R1        ;;POP STACK INTO R1
037004  012600                                 MOV     (SP)+,R0        ;;POP STACK INTO R0
037006  000240                                 NOP
037010  000207                                 RTS     PC
```

```
                                .SBTTL  GET SUBROUTINE

                                ;THIS SUBROUTINE READS THE REGISTERS WHICH ARE LISTED IN THE
                                ;"GET INDEX TABLE" AND STORES THEIR VALUES IN THE CORRESPONDING
                                ;LOCATION IN THE "GET REGISTER BUFFER".  FOR EXAMPLE, AN
                                ;ENTRY OF 04 IN THE TABLE WILL CAUSE THE SUBROUTINE TO
                                ;READ "RMBA" AND STORE ITS CONTENTS AT THE LOCATION IN
                                ;THE BUFFER ASSIGNED TO THAT REGISTER.  THE NUMBER OF
                                ;REGISTERS TO BE READ IS VARIABLE FROM 1 TO 22; THE INDEX
                                ;TABLE MUST BE TERMINATED WITH A CONTROL BYTE (200)
                                ;WHICH SHOULD FOLLOW THE LAST ENTRY.
                                ;
                                ;SUBROUTINE CALL:
                                ;(1)     "GET INDEX TABLE" HAS BEEN LOADED WITH REGISTER INDEX
                                ;       VALUES AND TERMINATED WITH A CONTROL BYTE
                                ;(2)     "GET INPUT BUFFER" IS AVAILABLE FOR USE.  (NOTE THAT
                                ;       UNUSED LOCATIONS, I.E.,ENTRIES IN BUFFER CORRESPONDING
                                ;       TO REGISTERS NOT READ, ARE NOT CHANGED.)
                                ;(3)     JSR     PC,GET
                                ;       BR      ???             RETURN HERE IF NO ERROR FOUND
                                ;       NOP                     RETURN HERE IF ANY ERROR FOUND
                                ;       ERROR                   SUB DEFINES ERROR NUMBER
                                ;       ???
                                ;
037012  000240          GET:    NOP
037014  062716  000004          ADD     #4,(SP)         ;CLEAR ERROR NUMBER IN USER'S
037020  105076  000000          CLRB    @(SP)           ;ERROR CALL
037024  162716  000004          SUB     #4,(SP)
037030  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
037032  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
037034  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
037036  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
037040  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
037042  013746  000004          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
037046  013746  000006          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
037052  013700  001276          MOV     $BASE,R0
037056  012702  001326          MOV     #GETBUF,R2
037062  012704  001504          MOV     #GETINX,R4
037066  012737  037174  000304  MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
037074  012737  000300  000006  MOV     #PR6,ERRVEC+2
037102  016037  000010  001174  1$:     MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
037110  016037  000000  001176          MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
037116  032737  004000  001176          BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
037124  001007                  BNE     3$              ;YES!!
037126  062766  000004  000016  ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN USER'S
037134  112776  000112  000016  MOVB    #112,@16(SP)    ;ERROR CALL
037142  000423                  BR      7$
037144  105714          3$:     TSTB    (R4)            ;DONE??
037146  100433                  BMI     9$              ;YES!!
037150  111401                  MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
037152  042701  177700          BIC     #^CIDXMSK,R1    ;CLEAR ANY SIGN EXTENSION
037156  060001                  ADD     R0,R1
037160  112403                  MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS FOR REGISTER
037162  042703  177700          BIC     #^CIDXMSK,R3    ;CLEAR ANY SIGN EXTENSION
037166  060203                  ADD     R2,R3
037170  011113                  MOV     (R1),(R3)       ;READ REGISTER
```

```
 8770  037172  000764                          BR      3$
 8771
 8772  037174  022626                  5$:     CMP     (SP)+,(SP)+     ;RESTORE STACK
 8773  037176  062766  000004  000016          ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
 8774  037204  112776  000007  000016          MOVB    #7,@16(SP)      ;USER'S ERROR CALL
 8775  037212  162766  000002  000016  7$:     SUB     #2,16(SP)
 8776  037220  105714                  8$:     TSTB    (R4)            ;DONE CLEARING??
 8777  037222  100405                          BMI     9$              ;YES!!
 8778  037224  005003                          CLR     R3              ;CLEAR REMAINING STORAGE
 8779  037226  112403                          MOVB    (R4)+,R3        ;LOCATIONS
 8780  037230  060203                          ADD     R2,R3
 8781  037232  005013                          CLR     (R3)
 8782  037234  000771                          BR      8$
 8783  037236                          9$:
 8784  037236  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
 8785  037242  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
 8786  037246  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
 8787  037250  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
 8788  037252  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 8789  037254  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 8790  037256  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 8791  037260  000207                          RTS     PC              ;RETURN
 8792
```

```
                                        PUT SUBROUTINE

                                .SBTTL  PUT SUBROUTINE

                                ;THIS SUBROUTINE WRITES THE REGISTERS WHICH ARE LISTED IN THE
                                ;"PUT INDEX TABLE" WITH  THE CONTENTS OF THE CORRESPONDING
                                ;LOCATION IN THE "PUT REGISTER BUFFER".  THE NUMBER OF
                                ;REGISTERS WRITTEN IS VARIABLE; THE INDEX TABLE MUST
                                ;BE TERMINATED WITH A CONTROL BYTE (200) WHICH SHOULD
                                ;FOLLOW THE LAST ENTRY.
                                ;
                                ;SUBROUTINE CALL:
                                ;
                                ;(1)     "PUT INDEX TABLE" HAS BEEN LOADED WITH INDEX VALUES
                                ;        OF REGISTERS TO BE WRITTEN.
                                ;(2)     "PUT REGISTER BUFFER" CONTAINS CONTENTS OF EACH
                                ;        REGISTER TO BE WRITTEN.
                                ;(3)     JSR     PC,PUT
                                ;        BR      ???             RETURN HERE IF NO ERROR FOUND
                                ;        NOP                     RETURN HERE IF ANY ERROR FOUND
                                ;        ERROR                   SUB DEFINES ERROR NUMBER
                                ;        ???
                                ;

037262  000240          PUT:    NOP
037264  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
037266  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
037270  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
037272  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
037274  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
037276  013746  000004          MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
037302  013746  000006          MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
037306  013700  001276          MOV     $BASE,R0
037312  012702  001376          MOV     #PUTBUF,R2
037316  012704  001533          MOV     #PUTINX,R4
037322  012737  037430  000004  MOV     #5$,ERRVEC      ;SETUP FOR TIMEOUT
037330  012737  000300  000006  MOV     #PR6,ERRVEC+2
037336  016037  000010  001174  1$:     MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
037344  016037  000000  001176          MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
037352  032737  004000  001176          BIT     #DVA,$TMP1      ;DEVICE AVAILABLE??
037360  001007                  BNE     3$              ;YES!!
037362  062766  000004  000016  ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
037370  112776  000112  000016  MOVB    #112,@16(SP)    ;USER'S ERROR CALL
037376  000423                  BR      7$
037400  105714          3$:     TSTB    (R4)            ;DONE??
037402  100424                  BMI     9$              ;YES!!
037404  111401                  MOVB    (R4),R1         ;R1 = REGISTER ADDRESS
037406  042701  177700          BIC     #^C1DXMSK,R1    ;CLEAR ANY SIGN EXTENSION
037412  060001                  ADD     R0,R1
037414  112403                  MOVB    (R4)+,R3        ;R3 = STORAGE ADDRESS
037416  042703  177700          BIC     #^C1DXMSK,R3    ;CLEAR ANY SIGN EXTENSION
037422  060203                  ADD     R2,R3
037424  011311                  MOV     (R3),(R1)       ;WRITE REGISTER
037426  000764                  BR      3$

037430  022626          5$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
037432  062766  000004  000016  ADD     #4,16(SP)       ;WRITE ERROR NUMBER IN
037440  112776  000007  000016  MOVB    #7,@16(SP)      ;USER'S ERROR CALL
037446  162766  000002  000016  7$:     SUB     #2,16(SP)
```

```
8849
8850  037454                          9$:
8851  037454  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
8852  037460  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
8853  037464  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
8854  037466  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
8855  037470  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
8856  037472  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
8857  037474  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8858  037476  000207                          RTS     PC              ;RETURN
8859
```

```
                                        .SBTTL  SIZE CLOCK SUBROUTINE

                                        SIZCLK:
                                                MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
                                                MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
                                                MOV     #1$,ERRVEC      ;SET UP FOR BUS TIMEOUT
                                                MOV     #PR6,ERRVEC+2
                                                MOV     #177546,CLKADR  ;LOAD ADDRESSES FOR KW11-L
                                                MOV     #100,CLKVCT
                                                TST     @CLKADR         ;TEST FOR KW11-L PRESENT
                                                BR      3$              ;YES - KW11-L IS PRESENT
                                        1$:     CMP     (SP)+,(SP)+     ;RESTORE SP
                                                MOV     #2$,ERRVEC      ;SET UP FOR BUS TIMEOUT
                                                MOV     #172540,CLKADR  ;LOAD ADDRESSES FOR KW11-P CLOCK
                                                MOV     #104,CLKVCT
                                                TST     @CLKADR         ;TEST FOR KW11-P PRESENT
                                                BR      3$              ;YES - KW11-P IS PRESENT
                                        2$:     CMP     (SP)+,(SP)+     ;RESTORE SP
                                                ADD     #2,4(SP)        ;MOVE RETURN TO ERROR
                                        3$:
                                                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
                                                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
                                                RTS     PC              ;RETURN TO USER
```

```
                                        .SBTTL  TIMEOUT SUBROUTINE
                                        ;
                                        ;THIS SUBROUTINE WAITS FOR GO TO RESET OR FOR A TIMEOUT GREATER THAN
                                        ;500 MS, WHICH EVER OCCURRS FIRST, AND THEN RETURNS.
                                        ;
                                        ;CALL:  JSR     PC,TIMOUT
                                        ;       ???                             RETURN HERE
                                        ;
037622                                  TIMOUT:
037622  010046                                MOV     R0,-(SP)        ;;PUSH R0 ON STACK
037624  010146                                MOV     R1,-(SP)        ;;PUSH R1 ON STACK
037626  010246                                MOV     R2,-(SP)        ;;PUSH R2 ON STACK
037630  013746  000004                        MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
037634  013746  000006                        MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
037640  012737  037742  000004                MOV     #4$,ERRVEC      ;SETUP FOR BUS TIMEOUT - 04 TRAP
037646  012737  000300  000006                MOV     #PR6,ERRVEC+2
037654  013700  001276                        MOV     $BASE,R0        ;R0=RM03 ADDRESS
037660  013701  001500                        MOV     CLKADR,R1       ;R1=CLOCK ADDRESS
037664  012702  000037                        MOV     #31.,R2         ;R2=NUMBER OF CLOCK CYCLES
037670  020127  172540          1$:           CMP     R1,#172540      ;KW11-P CLOCK??
037674  001003                                BNE     2$              ;NO!!
037676  012761  000001  000002                MOV     #1,2(R1)        ;SET COUNTER
037704  012711  000005          2$:           MOV     #BIT2!BIT0,(R1) ;START COUNTER
                                        ;
037710  016046  000000          3$:           MOV     RMCS1(R0),-(SP) ;GET STATUS
037714  042716  177576                        BIC     #^C<RDY!GO>,(SP)
037720  022726  000200                        CMP     #RDY,(SP)+      ;RDY=1,GO=0??
037724  001420                                BEQ     5$              ;YES!!
037726  032711  000200                        BIT     #BIT7,(R1)      ;TIMER DONE??
037732  001766                                BEQ     3$              ;NO!!
037734  005302                                DEC     R2              ;DEC NUMBER OF CYCLES
037736  001354                                BNE     1$              ;CONTINUE IF NOT DONE
037740  000412                                BR      5$
037742  022626                  4$:           CMP     (SP)+,(SP)+     ;ADJUST STACK
037744  062766  000004  000012                ADD     #4,12(SP)       ;MOVE SP TO USER'S CALL
037752  112776  000007  000012                MOVB    #7,@12(SP)      ;WRITE ERROR NUMBER
037760  162766  000002  000012                SUB     #2,12(SP)
                                        ;
037766                                  5$:
037766  012637  000006                        MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
037772  012637  000004                        MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
037776  012602                                MOV     (SP)+,R2        ;;POP STACK INTO R2
040000  012601                                MOV     (SP)+,R1        ;;POP STACK INTO R1
040002  012600                                MOV     (SP)+,R0        ;;POP STACK INTO R0
040004  000207                                RTS     PC              ;RETURN TO USER
                                        ;
```

```
                                .SBTTL  PRIMARY ERROR CHECK SUBROUTINE
                                ;
                                ;THE PURPOSE OF THIS SUBROUTINE IS TO VERIFY THAT STAUTS IS VALID AND
                                ;THAT FURTHER ERROR AND STATUS CHECKING SHOULD BE PERFORMED.  THE
                                ;FOLLOWING CHECKS ARE MADE:
                                ;
                                ;       .CURRECT UNIT IS SELECTED, I.E., THE UNIT SELECT BITS OF RMCS2
                                ;(BITS 0-2) EQUAL THE UNIT BEING TESTED;
                                ;
                                ;       .SELECTED UNIT IS AVAILABLE, I.E., DVA (BIT 11 OF RMCS1) IS SET
                                ;AND NED (BIT 12 OF RMCS2) IS RESET;
                                ;
                                ;       .LAST COMMAND WAS COMPLETED, I.E., THE MASSBUS CONTROLLER IS
                                ;READY (BIT 7 OF RMCS1) AND THE GO BIT IS RESET (BIT 0 OF RMCS1) OR THE
                                ;DRIVE READY BIT (BIT 7 OF RMDS) IS SET.
                                ;       .NO PARITY ERROR OCCURRED WHEN READING REMOTE REGISTERS,
                                ;I.E., MCPE = 0.
                                ;       NO PARITY ERROR OCCURRED WHEN WRITING REMOTE REGISTERS,
                                ;I.E., PAR = 0, OR, PAR = DPE = 1
                                ;
                                ;THE SUBROUTINE ASSUMES THAT:
                                ;
                                ;       .STATUS HAS BEEN STORED IN THE REGISTER INPUT BUFFER,
                                ;IN PARTICULAR, RMCS1, RMCS2 AND RMDS HAVE BEEN STORED IN THEIR
                                ;CORRESPONDING LOCATIONS OF THE "GET" BUFFER.
                                ;
                                ;       .($UNIT) CONTAINS THE DRIVE NUMBER
                                ;
                                ;THE SUBROUTINE IS CALLED AS FOLLOWS:
                                ;
                                ;(1)    JSR     PC,PRIERR
                                ;       BR      ???             RETURN HERE IF NO ERROR
                                ;       NOP                     RETURN HERE TO REPORT AN ERROR
                                ;       ERROR                   ERROR NUMBER DEFINED BY SUB
                                ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
                                ;       ???                     RETURN HERE IF NO MORE ERRORS
                                ;
040006                          PRIERR:
                                ;
                                ;CLEAR USER'S ERROR CALL
040006  062716  000004          ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
040012  105076  000000          CLRB    @(SP)           ;CLEAR ERROR NUMBER
040016  162716  000004          SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
                                ;
                                ;
                                ;REPORT AN ERROR IF THE WRONG UNIT IS SELECTED
040022  013737  001336  001142  MOV     RMCS2I,$BDDAT   ;CORRECT UNIT SELECTED??
040030  042737  177770  001142  BIC     #^CUNTMSK,$BDDAT
040036  013737  001234  001140  MOV     $UNIT,$GDDAT    ;GOOD DATA FOR TYPEOUT
040044  042737  177770  001140  BIC     #^CUNTMSK,$GDDAT
040052  123737  001140  001142  CMPB    $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
                                                        ;DRIVE NUMBERS
040060  001415                  BEQ     1$              ;YES!!
040062  062716  000004          ADD     #4,(SP)
040066  112776  000001  000000  MOVB    #1,@(SP)        ;ERROR 1
```

```
 8985  040074  162716  000002          SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
 8986  040100  004736                  JSR     PC,@(SP)+        ;REPORT WRONG UNIT SELECTED
 8987  040102  162716  000010          SUB     #10,(SP)         ;RESTORE (SP)
 8988  040106  000240                  NOP
 8989  040110  000137  040630          JMP     10$              ;SKIP OTHER CHECKS
 8990  040114                      1$:
 8991
 8992                                  ;REPORT AN ERROR IF THE DEVICE IS NOT AVAILABLE OR IF
 8993                                  ;THE DEVICE IS NONEXISTANT
 8994  040114  032737  004000  001326          BIT     #DVA,RMCS1I      ;DEVICE AVAILABLE??
 8995  040122  001045                  BNE     5$               ;YES!!
 8996  040124  013737  001326  001140          MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
 8997  040132  052737  004000  001140          BIS     #DVA,$GDDAT
 8998  040140  013737  001326  001142          MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
 8999  040146  062716  000004                  ADD     #4,(SP)
 9000  040152  112776  000002  000000          MOVB    #2,@(SP)         ;ERROR #2
 9001  040160  032737  010000  001336          BIT     #NED,RMCS2I      ;WAS NED SET??
 9002  040166  001414                  BEQ     2$               ;NO!!
 9003  040170  013737  001336  001140          MOV     RMCS2I,$GDDAT    ;EXPECTED STATUS
 9004  040176  013737  001336  001142          MOV     RMCS2I,$BDDAT    ;RECEIVED STATUS
 9005  040204  042737  010000  001140          BIC     #NED,$GDDAT
 9006  040212  112776  000003  000000          MOVB    #3,@(SP)         ;YES - CHANGE ERROR NUMBER
 9007  040220  162716  000002          2$:     SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
 9008  040224  004736                  JSR     PC,@(SP)+        ;REPORT DEVICE NOT AVAILABLE
 9009  040226  162716  000010          SUB     #10,(SP)         ;RESTORE (SP)
 9010  040232  000240                  NOP
 9011  040234  000575                  BR      10$              ;SKIP OTHER CHECKS
 9012  040236                      5$:
 9013
 9014                                  ;REPORT AN ERROR IF MASSBUS CONTROLLER IS NOT READY
 9015  040236  032737  000200  001326          BIT     #RDY,RMCS1I      ;CONTROLLER READY??
 9016  040244  001030                  BNE     7$               ;YES!!
 9017  040246  013737  001326  001140          MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
 9018  040254  052737  000200  001140          BIS     #RDY,$GDDAT
 9019  040262  042737  160001  001140          BIC     #SC!TRE!MCPE!GO,$GDDAT
 9020  040270  013737  001326  001142          MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
 9021  040276  062716  000004                  ADD     #4,(SP)
 9022  040302  112776  000004  000000          MOVB    #4,@(SP)         ;ERROR #4
 9023  040310  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
 9024  040314  004736                  JSR     PC,@(SP)+        ;REPORT CONTROLLER NOT READY
 9025  040316  162716  000010          SUB     #10,(SP)         ;RESTORE (SP)
 9026  040322  000240                  NOP
 9027  040324  000541                  BR      10$              ;SKIP OTHER CHECKS
 9028  040326                      7$:
 9029
 9030                                  ;REPORT AN ERROR IF GO IS NOT ZERO AND DRY IS NOT ONE
 9031  040326  032737  000001  001326          BIT     #GO,RMCS1I       ;GO RESET??
 9032  040334  001431                  BEQ     8$               ;YES!!
 9033  040336  032737  000200  001340          BIT     #DRY,RMDSI       ;DRIVE READY??
 9034  040344  001025                  BNE     8$               ;YES!!
 9035  040346  013737  001326  001140          MOV     RMCS1I,$GDDAT    ;EXPECTED STATUS
 9036  040354  042737  160001  001140          BIC     #SC!TRE!MCPE!GO,$GDDAT
 9037  040362  013737  001326  001142          MOV     RMCS1I,$BDDAT    ;RECEIVED STATUS
 9038  040370  062716  000004                  ADD     #4,(SP)
 9039  040374  112776  000005  000000          MOVB    #5,@(SP)         ;ERROR #5
 9040  040402  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
```

```
9041  040406  004736                        JSR     PC,@(SP)+       ;REPORT DRIVE NOT READY
9042  040410  162716  000010                SUB     #10,(SP)        ;RESTORE (SP)
9043  040414  000240                        NOP
9044  040416  000504                        BR      10$
9045  040420                          8$:
9046
9047                                  ;REPORT AN ERROR IF THE RH CONTROLLER DETECTED BAD
9048                                  ;PARITY ON THE MASSBUS CONTROL BUS
9049  040420  032737  020000  001326        BIT     #MCPE,RMCS1I    ;PARITY ERROR ??
9050  040426  001425                        BEQ     9$              ;NO!!
9051  040430  013737  001326  001140        MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
9052  040436  042737  160001  001140        BIC     #SC!TRE!MCPE!GO,$GDDAT
9053  040444  013737  001326  001142        MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
9054  040452  062716  000004                ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
9055  040456  112776  000013  000000        MOVB    #13,@(SP)       ;ERROR #47
9056  040464  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
9057  040470  004736                        JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
9058  040472  162716  000010                SUB     #10,(SP)        ;RESTORE STACK
9059  040476  000240                        NOP
9060  040500  000453                        BR      10$
9061  040502                          9$:
9062
9063                                  ;REPORT AN ERROR IF THE RM03 DETECTED A CONTROL BUS PARITY ERROR
9064  040502  032737  000010  001342        BIT     #PAR,RMER1I     ;WAS THERE A PARITY ERROR??
9065  040510  001451                        BEQ     11$             ;NO!!
9066  040512  032737  000010  001370        BIT     #DPE,RMER2I     ;WAS IT THE CONTROL BUS??
9067  040520  001045                        BNE     11$             ;NOT SURE!!
9068  040522  032737  000010  001412        BIT     #PAR,RMER1O     ;DID TEST SET PAR ??
9069  040530  001413                        BEQ     93$             ;NO!!
9070  040532  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
9071  040534  012700  001533                MOV     #PUTINX,R0      ;R0 POINTS TO INDEX TABLE
9072  040540  122710  000014          91$:  CMPB    #RMER1,(R0)     ;SEARCH TABLE FOR RMER1
9073  040544  001002                        BNE     92$
9074  040546  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
9075  040550  000431                        BR      11$             ;PAR WAS SET BY TEST
9076  040552  105720                  92$:  TSTB    (R0)+           ;END OF TABLE??
9077  040554  100371                        BPL     91$             ;NO!!
9078  040556  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
9079  040560  013737  001342  001140  93$:  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
9080  040566  042737  000010  001140        BIC     #PAR,$GDDAT
9081  040574  013737  001342  001142        MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
9082  040602  062716  000004                ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
9083  040606  112776  000050  000000        MOVB    #50,@(SP)       ;WRITE THE ERROR NUMBER
9084  040614  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
9085  040620  004736                        JSR     PC,@(SP)+       ;REPORT THE ERROR
9086  040622  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
9087  040626  000240                        NOP
9088  040630  062716  000010          10$:  ADD     #10,(SP)                ;RETURN TO ERROR
9089  040634  000240                  11$:  NOP                     ;RETURN TO NO ERROR
9090  040636  000207                        RTS     PC
9091
```

```
                                .SBTTL  SECONDARY ERROR CHECK SUBROUTINE
                                ;
                                ;THE ERROR CHECK SUBROUTINE PROVIDES DETECTION OF SECONDARY ERRORS
                                ;SUCH AS UNEXPECTED ERRORS AND UNEXPECTED REGISTER
                                ;CONTENTS.  THESE ERRORS ARE CONSIDERED
                                ;SECONDARY IN THAT THEY ARE NOT NECESSARILY ASSOCIATED WITH THE OPERATION
                                ;BEING PERFORMED.  WHEN THE SUBROUTINE IDENTIFIES SUCH AN ERROR, IT MOVES
                                ;THE ERROR NUMBER TO THE ERROR CALL IN THE TEST ROUTINE AND THEN RETURNS
                                ;TO THE TEST ROUTINE WHICH MAKES THE ERROR CALL.  AFTER THE TEST ROUTINE
                                ;MAKES THE ERROR CALL, IT RETURNS TO THE SUBROUTINE WHICH THEN LOOKS FOR
                                ;OTHER ERRORS.  WHEN ALL ERRORS HAVE BEEN REPORTED, THE SUBROUTINE
                                ;RETURNS TO THE ADDRESS FOLLOWING THE SUBROUTINE CALL.
                                ;
                                ;CALL:  JSR     PC,SECERR
                                ;       BR      ???             RETURN HERE IF NO ERROR
                                ;       NOP                     RETURN HERE TO REPORT AN ERROR
                                ;       ERROR                   ERROR NUMBER DEFINED BY SUB
                                ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
                                ;       ???                     RETURN HERE IF NO MORE ERRORS
                                ;
                                ;NOTE:  THE SUBROUTINE ASSUMES THAT REGISTERS HAVE BEEN STORED AT THE
                                ;INPUT REGISTER BUFFER.
                                ;
040640                          SECERR:
                                ;
                                ;;*****************************************************************
                                ;STORE FUNCTION CODE AND CLEAR USER'S ERROR NUMBER
040640  013737  001376  044500          MOV     RMCS1O,5I5$     ;STORE FUNCTION CODE
040646  042737  177701  044500          BIC     #^C<FO!F1!F2!F3!F4>,5I5$
040654  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
040660  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
040664  162716  000004                  SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
                                ;
                                ;;*****************************************************************
                                ;CHECK SECONDARY ERRORS COMMON TO ALL COMMANDS
                                ;
                                ;REPORT ERROR IF DRIVE IS NOT READY, I.E., IF DRY = 0
040670  032737  000200  001340          BIT     #DRY,RMDSI      ;DRIVE READY??
040676  001024                  BNE     5$                      ;YES!!
040700  013737  001340  001142          MOV     RMDSI,$BDDAT    ;BAD DATA FOR TYPEOUT
040706  042737  177577  001142          BIC     #^CDRY,$BDDAT
040714  012737  000200  001140          MOV     #DRY,$GDDAT     ;GOOD DATA FOR TYPEOUT
040722  062716  000004                  ADD     #4,(SP)
040726  112776  000010  000000          MOVB    #10,@(SP)       ;ERROR NUMBER
040734  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
040740  004736                  JSR     PC,@(SP)+       ;REPORT NOT READY
040742  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP) TO ERROR N
040746  000240                  NOP
                                ;
                                ;REPORT ERROR IF GO BIT IS NOT RESET
040750  032737  000001  001326  5$:     BIT     #GO,RMCS1I      ;GO BIT RESET??
040756  001423                  BEQ     10$                     ;YES!!
040760  013737  001326  001142          MOV     RMCS1I,$BDDAT   ;BAD DATA FOR TYPEOUT
040766  042737  177776  001142          BIC     #^CGO,$BDDAT
040774  005037  001140                  CLR     $GDDAT          ;GOOD DATA FOR TYPEOUT
041000  062716  000004                  ADD     #4,(SP)
```

```
9148  041004  112776  000011  000000          MOVB    #11,@(SP)       ;ERROR NUMBER
9149  041012  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
9150  041016  004736                          JSR     PC,@(SP)+       ;REPORT DEVICE NOT AVAILABLE
9151  041020  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
9152  041024  000240                          NOP
9153
9154                                  ;REPORT ERROR IF FUNCTION CODE READ FROM DEVICE IS NOT CORRECT
9155  041026  013737  001326  001142  10$:     MOV     RMCS1I,$BDDAT   ;IS FUNCTION CODE CORRECT??
9156  041034  042737  177701  001142          BIC     #^C76,$BDDAT
9157  041042  013737  044500  001140          MOV     515$,$GDDAT     ;EXPECTED FUNCTION CODE
9158  041050  023737  001142  001140          CMP     $BDDAT,$GDDAT
9159  041056  001413                          BEQ     15$             ;YES!!
9160  041060  062716  000004                  ADD     #4,(SP)
9161  041064  112776  000012  000000          MOVB    #12,@(SP)       ;ERROR NUMBER
9162  041072  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
9163  041076  004736                          JSR     PC,@(SP)+       ;REPORT WRONG FUNCTION CODE
9164  041100  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
9165  041104  000240                          NOP
9166  041106                          15$:
9167                                  ;REPORT AN ERROR IF COMPOSITE ERROR IS SET AND NO OTHER
9168                                  ;ERRORS ARE SET, OR IF COMPOSITE ERROR IS NOT SET AND
9169                                  ;OTHER ERRORS ARE SET
9170  041106  005037  001140                  CLR     $GDDAT          ;EXPECT "ERR"=0
9171  041112  005737  001342                  TST     RMER1I          ;IS RMER1 =0??
9172  041116  001003                          BNE     20$             ;NO!!
9173  041120  005737  001370                  TST     RMER2I          ;IS RMERZ=0??
9174  041124  001403                          BEQ     25$             ;YES!!
9175  041126  052737  040000  001140  20$:     BIS     #ERR,$GDDAT     ;"ERR" SOULD BE SET
9176  041134  013737  001340  001142  25$:     MOV     RMDS1,$BDDAT
9177  041142  042737  137777  001142          BIC     #^CERR,$BDDAT
9178  041150  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "ERR" OK??
9179  041156  001412                          BEQ     30$             ;YES!!
9180  041160  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
9181  041164  112776  000047  000000          MOVB    #47,@(SP)       ;WRITE ERROR NUMBER
9182  041172  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9183  041176  004736                          JSR     PC,@(SP)+       ;REPORT INVALID COMP ERROR
9184  041200  162716  000010                  SUB     #10,(SP)
9185
9186                                  ;REPORT AN ERROR IF "TRE" IS SET AND NONE OF THE BITS WHICH SET
9187                                  ;TRE IS SET, OR IF TRE IS NOT SET AND ONE OR MORE BITS WHICH
9188                                  ;SET TRE IS SET
9189  041204  005037  001140          30$:    CLR     $GDDAT          ;EXPECT "TRE" =0
9190  041210  013746  001336                  MOV     RMCS2I,-(SP)    ;WAS DLT, WCE, UPE, NED, NEM
9191  041214  042726  000377                  BIC     #377,(SP)+      ;PGE, MXF OR MDPE SET
9192  041220  001010                          BNE     35$             ;YES!!
9193  041222  032737  040000  001340          BIT     #ERR,RMDSI      ;WAS EXCEPTION RECEIVED??
9194  041230  001407                          BEQ     40$             ;NO!!
9195  041232  022737  000030  044500          CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
9196  041240  103003                          BHIS    40$             ;NO!!
9197  041242  052737  040000  001140  35$:     BIS     #TRE,$GDDAT     ;"TRE" SHOULD BE SET
9198  041250  013737  001326  001142  40$:     MOV     RMCS1I,$BDDAT   ;BAD DATA FOR TYPEOUT
9199  041256  042737  137777  001142          BIC     #^CTRE,$BDDAT
9200  041264  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "TRE" OK??
9201  041272  001413                          BEQ     45$             ;YES!!
9202  041274  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
9203  041300  112776  000014  000000          MOVB    #14,@(SP)       ;WRITE ERROR NUMBER
```

```
 9204  041306  162716  000002                SUB     #2,(SP)          ;MOVE SP TO RETURN FOR ERROR
 9205  041312  004736                        JSR     PC,@(SP)+        ;REPORT TRE ERROR
 9206  041314  162716  000010                SUB     #10,(SP)         ;RESTORE (SP)
 9207  041320  000240                        NOP
 9208  041322                          45$:
 9209
 9210                                  ;;*****************************************************************
 9211                                  ;USING THE FUNCTION CODE TABLE, CHECK FOR THE FOLLOWING ERRORS:
 9212                                  ;       .STATUS BITS NOT SET THAT SHOULD BE SET, E.G., ATA AND ILF
 9213                                  ;       .STATUS BITS SET THAT SHOULD NOT BE SET, E.G., WCE AND ECH
 9214                                  ;NOTE THAT SOME ERROR BITS ARE CONDITIONAL ON THE COMMAND AND OTHER
 9215                                  ;STATUS CONDITIONS, E.G., WRITE LOCK ERROR SHOULD ONLY BE SET IF
 9216                                  ;WRITE LOCK IS ON AND THE COMMAND IS A WRITE.
 9217
 9218                                  ;GET AND STORE THE ENTRY FROM THE FUNCTION CODE TABLE
 9219  041322  010046                        MOV     R0,-(SP)         ;;PUSH R0 ON STACK
 9220  041324  013700  044500                MOV     515$,R0          ;R0 = FUNCTION CODE
 9221  041330  016037  063676  044472        MOV     FNCDTB(R0),500$  ;STORE ENTRY
 9222  041336  012600                        MOV     (SP)+,R0         ;;POP STACK INTO R0
 9223
 9224                                  ;REPORT AN ERROR IF AN UNEXPECTED ATTENTION OCCURRED OR IF
 9225                                  ;ATA IS NOT SET AND SHOULD BE SET.
 9226  041340  013737  044472  001140        MOV     500$,$GDDAT      ;GET EXPECTED ATA STATUS
 9227  041346  032737  040000  001340        BIT     #ERR,RMDSI       ;IS COMPOSITE ERROR SET ??
 9228  041354  001403                        BEQ     50$              ;NO !!
 9229  041356  052737  100000  001140        BIS     #ATA,$GDDAT      ;EXPECT AN ATTENTION
 9230  041364  042737  077777  001140  50$:   BIC     #^CATA,$GDDAT    ;STRIP DONT CARES
 9231  041372  013737  001340  001142        MOV     RMDSI,$BDDAT     ;GET RECEIVED ATA
 9232  041400  042737  077777  001142        BIC     #^CATA,$BDDAT    ;STRIP DONT CARES
 9233  041406  023737  001140  001142        CMP     $GDDAT,$BDDAT    ;IS ATA OK ??
 9234  041414  001413                        BEQ     55$              ;YES !!
 9235  041416  062716  000004                ADD     #4,(SP)          ;MOVE SP TO USERS ERROR CALL
 9236  041422  112776  000006  000000        MOVB    #6,@(SP)         ;LOAD ERROR # IN CALL
 9237  041430  162716  000002                SUB     #2,(SP)          ;MOVE SP TO ERROR RETURN
 9238  041434  004736                        JSR     PC,@(SP)+        ;REPORT ERROR
 9239  041436  162716  000010                SUB     #10,(SP)         ;RESTORE SP
 9240  041442  000240                        NOP
 9241  041444                          55$:
 9242
 9243                                  ;REPORT ERROR IF ILF IS INCORRECT, I.E., IF ILF DOES NOT COMPARE
 9244                                  ;WITH FUNCTION CODE TABLE
 9245  041444  013737  044472  001140        MOV     500$,$GDDAT      ;GET EXPECTED ILF
 9246  041452  042737  177776  001140        BIC     #^CILF,$GDDAT    ;CLEAR ALL OTHER BITS
 9247  041460  013737  001342  001142        MOV     RMER1I,$BDDAT    ;GET RECEIVED ILF
 9248  041466  042737  177776  001142        BIC     #^CILF,$BDDAT    ;CLEAR ALL OTHER BITS
 9249  041474  023737  001140  001142        CMP     $GDDAT,$BDDAT    ;IS ILF OK ??
 9250  041502  001412                        BEQ     60$              ;YES !!
 9251  041504  062716  000004                ADD     #4,(SP)          ;MOVE SP TO USERS ERROR CALL
 9252  041510  112776  000254  000000        MOVB    #254,@(SP)       ;WRITE ERROR NUMBER IN CALL
 9253  041516  162716  000002                SUB     #2,(SP)          ;MOVE SP TO ERROR RETURN
 9254  041522  004736                        JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
 9255  041524  162716  000010                SUB     #10,(SP)         ;MOVE SP TO NO ERROR
 9256  041530  005037  001140          60$:   CLR     $GDDAT           ;CLEAR EXPECTED STATUS
 9257
 9258                                  ;REPORT AN ERROR IF WCE IS SET AND SHOULD NOT BE SET
 9259  041534  013746  044472                MOV     500$,-(SP)       ;GET WCE STATUS ENABLE
```

```
                        BIS     #↑CWCE,(SP)     ;SET ALL OTHER BITS
                        MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
                        BIC     (SP)+,$BDDFT    ;CLEAR WCE IF ENABLED
                        BEQ     90$             ;BRANCH IF WCE OK
                        ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
                        MOVB    #26,@(SP)       ;WRITE ERROR NUMBER
                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
                        JSR     PC,@(SP)+       ;REPORT ERROR
                        SUB     #10,(SP)        ;RESTORE ERROR
                90$:

                ;REPORT ERROR IF OPI STATUS IS SET AND SHOULD NOT BE SET
                        MOV     500$,-(SP)      ;GET OPI STATUS ENABLE
                        BIS     #↑COPI,(SP)     ;SET ALL OTHER BITS
                        MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
                        BIC     (SP)+,$BDDAT    ;CLEAR OPI IF ENABLED
                        BEQ     100$            ;BRANCH IF OPI OK
                        ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
                        MOVB    #164,@(SP)      ;WRITE ERROR NUMBER IN CALL
                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
                        JSR     PC,@(SP)+       ;REPORT ERROR
                        SUB     #10,(SP)        ;RESTORE SP
                100$:

                ;REPORT ERROR IF IVC IS SET AND IS NOT ENABLED OR IF IVC IS
                ;SET AND VV IS NOT RESET
                        MOV     500$,-(SP)      ;GET IVC STATUS ENABLE
                        BIT     #VV,RMDSI       ;IS VV SET
                        BEQ     105$            ;NO !!
                        BIC     #IVC,(SP)       ;YES - IVC SHOULD BE 0
                105$:   BIS     #↑CIVC,(SP)     ;SET ALL OTHER BITS
                        MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
                        BIC     (SP)+,$BDDAT    ;CLEAR IVC IF ENABLED
                        BEQ     110$            ;BRANCH IF IVC OK
                        ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
                        MOVB    #165,@(SP)      ;WRITE ERROR NUMBER IN CALL
                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
                        JSR     PC,@(SP)+       ;REPORT ERROR
                        SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
                110$:

                ;BIT 11 (WLE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
                ; ALL WRITE ERRORS, I.E.,
                ;       RMER1 - WLE, WCF
                ;       RMER2 - DPE
                ;       RMCS2 - UPE.
                ;EACH OF THESE ERRORS IS CHECKED TO SEE IF AN ERROR IS SET WHEN THE
                ;WRITE ERROR ENABLE BIT IS RESET.

                ;REPORT AN ERROR IF WLE IS SET AND WRITE ERRORS ARE NOT ENABLED, OR IF
                ;THE DRIVE IS NOT WRITE PROTECTED
                        MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ENABLED
                        BIT     #WLE,500$       ;ARE WRITE ERRORS ENABLED ??
                        BEQ     115$            ;NO !!
                        BIT     #WRL,RMDSI      ;IS THE DRIVE WRITE PROTECTED ??
                        BNE     120$            ;YES !!
```

```
 9316  041764  042716  004000          115$:   BIC     #WLE,(SP)       ;RESET WLE ENABLE
 9317  041770  013737  001342  001142  120$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
 9318  041776  042637  001142                  BIC     (SP)+,$BDDAT    ;CLEAR WLE IF ENABLED
 9319  042002  001412                          BEQ     125$            ;BRANCH IF WLE OK
 9320  042004  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 9321  042010  112776  000023  000000          MOVB    #23,@(SP)       ;WRITE ERROR NUMBER IN CALL
 9322  042016  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9323  042022  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
 9324  042024  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
 9325  042030                          125$:
 9326
 9327                                  ;REPORT ERROR IF WCF IS SET AND WRITE ERRORS ARE NOT ENABLED
 9328  042030  012746  177777                  MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ENABLED
 9329  042034  032737  004000  044472          BIT     #WLE,$OOS       ;ARE WRITE ERRORS ENABLED ??
 9330  042042  001002                          BNE     130$            ;YES !!
 9331  042044  042716  000040                  BIC     #WCF,(SP)       ;DISABLE WCF ERROR
 9332  042050  013737  001342  001142  130$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
 9333  042056  042637  001142                  BIC     (SP)+,$BDDAT    ;RESET WCF IF ENABLED
 9334  042062  001412                          BEQ     135$            ;BRANCH IF WCF OK
 9335  042064  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 9336  042070  112776  000025  000000          MOVB    #25,@(SP)       ;WRITE ERROR NUMBER IN CALL
 9337  042076  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9338  042102  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
 9339  042104  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
 9340  042110                          135$:
 9341
 9342                                  ;REPORT ERROR IF DPE IS SET AND WRITE ERRORS ARE NOT ENABLED
 9343  042110  012746  177777                  MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ARE ENABLED
 9344  042114  032737  004000  044472          BIT     #WLE,$OOS       ;ARE WRITE ERRORS ENABLED ??
 9345  042122  001002                          BNE     140$            ;YES !!
 9346  042124  042716  000010                  BIC     #DPE,(SP)       ;RESET DPE ENABLE
 9347  042130  013737  001370  001142  140$:   MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
 9348  042136  042637  001142                  BIC     (SP)+,$BDDAT    ;RESET DPE IF ENABLED
 9349  042142  001412                          BEQ     145$            ;BRANCH IF DPE OK
 9350  042144  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 9351  042150  112776  000040  000000          MOVB    #40,@(SP)       ;WRITE ERROR NUMBER IN CALL
 9352  042156  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9353  042162  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
 9354  042164  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
 9355  042170                          145$:
 9356
 9357                                  ;REPORT AN ERROR IF UPE IS SET AND WRITE ERRORS ARE NOT ENABLED
 9358  042170  012746  177777                  MOV     #-1,-(SP)       ;ASSUME WRITE ERRORS ARE ENABLED
 9359  042174  032737  004000  044472          BIT     #WLE,$OOS       ;ARE WRITE ERRORS ENABLED ??
 9360  042202  001002                          BNE     150$            ;YES !!
 9361  042204  042716  020000                  BIC     #UPE,(SP)       ;DISABLE UPE ERROR
 9362  042210  013737  001336  001142  150$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
 9363  042216  042637  001142                  BIC     (SP)+,$BDDAT    ;RESET UPE IF ENABLED
 9364  042222  001412                          BEQ     155$            ;BRANCH IF UPE OK
 9365  042224  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 9366  042230  112776  000024  000000          MOVB    #24,@(SP)       ;WRITE ERROR NUMBER IN CALL
 9367  042236  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9368  042242  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
 9369  042244  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
 9370  042250                          155$:
 9371
```

```
 9372                                   ;REPORT AN ERROR IF IAE IS SET AND IS NOT ENABLED
 9373  042250  013746  044472                MOV     500$,-(SP)      ;GET IAE ENABLE
 9374  042254  052716  175777                BIS     #^C<IAE>,(SP)   ;SET ALL OTHER BITS
 9375  042260  013737  001342  001142        MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
 9376  042266  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR IAE IF ENABLED
 9377  042272  001412                        BEQ     160$            ;BRANCH IF IAE IS OK
 9378  042274  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 9379  042300  112776  000166  000000        MOVB    #166,@(SP)      ;WRITE ERROR NUMBER
 9380  042306  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9381  042312  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
 9382  042314  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
 9383  042320                         160$:
 9384
 9385                                   ;BIT 09 (AOE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
 9386                                   ; ALL READ/WRITE ERRORS, I.E.,
 9387                                   ;       RMCS1 - TRE
 9388                                   ;       RMCS2 - DLT,NEM,MXF
 9389                                   ;       RMDS  - LBT
 9390                                   ;       RMER1 - AOE
 9391                                   ;NOTE:
 9392                                   ;       LBT IS NOT CHECKED BECAUSE IT ONLY RESETS WHEN THE DESIRED
 9393                                   ;CYLINDER REGISTER IS WRITTEN
 9394                                   ;NOTE:
 9395                                   ;       AOE CANNOT BE SET IF LBT IS NOT ALSO SET
 9396                                   ;NOTE:
 9397                                   ;       TRE IS CHECKED AS A FUNCTION OF OTHER ERROR CONDITONS ABOVE
 9398
 9399
 9400                                   ;REPORT AN ERROR IF DLT IS SET AND READ/WRITE ERRORS ARE NOT E'ABLED
 9401  042320  012746  177777                MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
 9402  042324  032737  001000  044472        BIT     #AOE,500$       ;ARE ERRORS ENABLED ??
 9403  042332  001002                        BNE     165$            ;YES !!
 9404  042334  042716  100000                BIC     #DLT,(SP)       ;RESET DLT ENABLE
 9405  042340  013737  001336  001142  165$: MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
 9406  042346  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR DLT IF ENABLED
 9407  042352  001412                        BEQ     170$            ;BRANCH IF DLT IS OK
 9408  042354  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 9409  042360  112776  000032  000000        MOVB    #32,@(SP)       ;WRITE ERROR NUMBER IN CALL
 9410  042366  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9411  042372  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
 9412  042374  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
 9413  042400                         170$:
 9414
 9415                                   ;REPORT ERROR IF NEM IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
 9416  042400  012746  177777                MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
 9417  042404  032737  001000  044472        BIT     #AOE,500$       ;ARE ERRORS ENABLED ??
 9418  042412  001002                        BNE     175$            ;YES !!
 9419  042414  042716  004000                BIC     #NEM,(SP)       ;DISABLE NEM
 9420  042420  013737  001336  001142  175$: MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
 9421  042426  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR NEM IF ENABLED
 9422  042432  001412                        BEQ     180$            ;BRANCH IF NEM IS OK
 9423  042434  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 9424  042440  112776  000167  000000        MOVB    #167,@(SP)      ;WRITE ERROR NUMBER IN CALL
 9425  042446  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9426  042452  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
 9427  042454  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
```

```
 9428  042460                                  180$:
 9429
 9430                                          ;REPORT ERROR IF MXF IS SET AND READ/WRITE ERRORS ARE NOT ENABLED
 9431  042460  012746  177777                        MOV     #-1,-(SP)       ;ASSUME ERRORS ARE ENABLED
 9432  042464  032737  001000  044472                BIT     #AOE,500$       ;ARE DATA ERRORS ENABLED ??
 9433  042472  001002                                BNE     185$            ;YES !!
 9434  042474  042716  001000                        BIC     #MXF,(SP)       ;DISABLE MXF ERROR
 9435  042500  013737  001336  001142  185$:         MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
 9436  042506  042637  001142                        BIC     (SP)+,$BDDAT    ;CLEAR MXF IF ENABLED
 9437  042512  001412                                BEQ     190$            ;BRANCH IF MXF IS OK
 9438  042514  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 9439  042520  112776  000033  000000                MOVB    #33,@(SP)       ;WRITE ERROR NUMBER IN CALL
 9440  042526  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9441  042532  004736                                JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
 9442  042534  162716  000010                        SUB     #10,(SP)        ;MOVE SP TO NO ERROR
 9443  042540                                  190$:
 9444
 9445                                          ;REPORT ERROR IF AOE IS SET AND DATA ERRORS ARE NOT ENABLED
 9446  042540  012746  177777                        MOV     #-1,-(SP)       ;ASSUME DATA ERRORS ARE ENABLED
 9447  042544  032737  001000  044472                BIT     #AOE,500$       ;ARE DATA ERRORS ENABLED ??
 9448  042552  001404                                BEQ     191$            ;NO !!
 9449  042554  032737  002000  001340                BIT     #LBT,RMDSI      ;IS LBT ALSO SET ??
 9450  042562  001002                                BNE     195$            ;YES !!
 9451  042564  042716  001000  191$:                 BIC     #AOE,(SP)       ;DISABLE AOE
 9452  042570  013737  001342  001142  195$:         MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
 9453  042576  042637  001142                        BIC     (SP)+,$BDDAT    ;CLEAR AOE IF ENABLED
 9454  042602  001412                                BEQ     200$            ;BRANCH IF AOE IS OK
 9455  042604  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 9456  042610  112776  000020  000000                MOVB    #20,@(SP)       ;WRITE ERROR NUMBER
 9457  042616  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9458  042622  004736                                JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
 9459  042624  162716  000010                        SUB     #10,(SP)        ;MOVE SP TO NO ERROR
 9460  042630                                  200$:
 9461
 9462                                          ;BIT 07 (HCE) OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR
 9463                                          ;HEADER ERRORS, I.E.,
 9464                                          ;       RMER1 - HCRC,HCE,FER
 9465                                          ;       RMER2 - BSE
 9466
 9467                                          ;RESET THE ENABLING BIT (HCE) IF HEADER COMPARE INHIBIT IS SET
 9468  042630  032737  002000  001360                BIT     #HCI,RMOFI      ;IS HCI SET ??
 9469  042636  001403                                BEQ     201$            ;NO !!
 9470  042640  042737  000200  044472                BIC     #HCE,500$       ;YES - DISABLE ALL HEADER ERRORS
 9471  042646                                  201$:
 9472
 9473                                          ;REPORT AN ERROR IF HCRC IS SET AND HEADER ERRORS ARE NOT ENABLED
 9474  042646  012746  177777                        MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
 9475  042652  032737  000200  044472                BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
 9476  042660  001002                                BNE     205$            ;YES !!
 9477  042662  042716  000400                        BIC     #HCRC,(SP)      ;DISABLE HCRC
 9478  042666  013737  001342  001142  205$:         MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
 9479  042674  042637  001142                        BIC     (SP)+,$BDDAT    ;RESET HCRC IF ENABLED
 9480  042700  001412                                BEQ     210$            ;BRANCH IF HCRC IS OK
 9481  042702  062716  000004                        ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 9482  042706  112776  000035  000000                MOVB    #35,@(SP)       ;WRITE ERROR NUMBER IN CALL
 9483  042714  162716  000002                        SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
```

```
9484  042720  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9485  042722  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9486  042726                        210$:
9487
9488                                ;REPORT ERROR IF HCE IS SET AND HEADER ERRORS ARE NOT ENABLED
9489  042726  012746  177777                MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
9490  042732  032737  000200  044472        BIT     #HCE,500$       ;ARE ERRORS ENABLED ??
9491  042740  001002                        BNE     215$            ;YES !!
9492  042742  042716  000200                BIC     #HCE,(SP)       ;DISABLE HCE
9493  042746  013737  001342  001142  215$: MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
9494  042754  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR HCE IF ENABLED
9495  042760  001412                        BEQ     220$            ;BRANCH IF HCE IS OK
9496  042762  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9497  042766  112776  000036  000000        MOVB    #36,@(SP)       ;WRITE ERROR NUMBER IN CALL
9498  042774  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9499  043000  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9500  043002  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9501  043006                        220$:
9502
9503                                ;REPORT ERROR IF FER IS SET AND HEADER ERRORS ARE NOT ENABLED
9504  043006  012746  177777                MOV     #-1,-(SP)       ;ASSUME FER IS ENABLED
9505  043012  032737  000200  044472        BIT     #HCE,500$       ;ARE HEADER ERRORS ENABLED ??
9506  043020  001002                        BNE     225$            ;YES !!
9507  043022  042716  000020                BIC     #FER,(SP)       ;DISABLE FER
9508  043026  013737  001342  001142  225$: MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
9509  043034  042637  001142                BIC     (SP)+,$BDDAT    ;RESET FER IF ENABLED
9510  043040  001412                        BEQ     230$            ;BRANCH IF FER OK
9511  043042  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9512  043046  112776  000037  000000        MOVB    #37,@(SP)       ;WRITE ERROR NUMBER IN CALL
9513  043054  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9514  043060  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9515  043062  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9516  043066                        230$:
9517
9518                                ;REPORT ERROR IF BSE IS SET AND HEADER ERRORS ARE NOT ENABLED
9519  043066  012746  177777                MOV     #-1,-(SP)       ;ASSUME ERRORS ENABLED
9520  043072  032737  000200  044472        BIT     #HCE,500$       ;ARE THEY ENABLED ??
9521  043100  001002                        BNE     235$            ;YES !!
9522  043102  042716  100000                BIC     #BSE,(SP)       ;DISABLE BSE
9523  043106  013737  001370  001142  235$: MOV     RMER2I,$BDDAT   ;GET RECEIVED STATUS
9524  043114  042637  001142                BIC     (SP)+,$BDDAT    ;CLEAR BSE IF ENABLED
9525  043120  001412                        BEQ     240$            ;BRANCH IF BSE OK
9526  043122  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9527  043126  112776  000354  000000        MOVB    #354,@(SP)      ;WRITE ERROR NUMBER
9528  043134  162716  000002                SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9529  043140  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9530  043142  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9531  043146                        240$:
9532
9533                                ;BIT 06 OF THE FUNCTION CODE TABLE IS THE ENABLING BIT FOR DATA
9534                                ;FIELD ERRORS, I.E.,
9535                                ;       RMCS2 - MDPE
9536                                ;       RMER1 - DCK,ECH
9537                                ;NOTE:
9538                                ;       ECH CANNOT SET UNLESS IT IS ENABLED AND ECI IS RESET AND
9539                                ;DCK IS SET.
```

```
9540
9541                                    ;REPORT ERROR IF MDPE IS SET AND IS NOT ENABLED
9542  043146  012746  177777            MOV     #-1,-(SP)       ;ASSUME ENABLED
9543  043152  032737  000100  044472    BIT     #ECH,5OO$       ;ARE DATA FIELD ERRORS ENABLED ??
9544  043160  001002                    BNE     245$            ;YES !!
9545  043162  042716  000400            BIC     #MDPE,(SP)      ;DISBALE MDPE
9546  043166  013737  001336  001142  245$:   MOV     RMCS2I,$BDDAT   ;GET RECEIVED STATUS
9547  043174  042637  001142            BIC     (SP)+,$BDDAT    ;CLEAR MDPE IF ENABLED
9548  043200  001412                    BEQ     250$            ;BRANCH IF MDPE OK
9549  043202  062716  000004            ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9550  043206  112776  000027  000000    MOVB    #27,@(SP)       ;WRITE ERROR NUMBER IN CALL
9551  043214  162716  000002            SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9552  043220  004736                    JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9553  043222  162716  000010            SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9554  043226                          250$:
9555
9556                                    ;REPORT ERROR IF DCK IS SET AND DATA FIELD ERRORS ARE NOT ENABLED
9557  043226  012746  177777            MOV     #-1,-(SP)       ;ASSUME ENABLED
9558  043232  032737  000100  044472    BIT     #ECH,5OO$       ;ARE THEY ENABLED ??
9559  043240  001002                    BNE     255$            ;YES !!
9560  043242  042716  100000            BIC     #DCK,(SP)       ;DISABLE DCK
9561  043246  013737  001342  001142  255$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
9562  043254  042637  001142            BIC     (SP)+,$BDDAT    ;CLEAR DCK IF ENABLED
9563  043260  001412                    BEQ     260$            ;BRANCH IF DCK IS OK
9564  043262  062716  000004            ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9565  043266  112776  000030  000000    MOVB    #30,@(SP)       ;WRITE ERROR NUMBER IN CALL
9566  043274  162716  000002            SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9567  043300  004736                    JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9568  043302  162716  000010            SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9569  043306                          260$:
9570
9571                                    ;REPORT ERROR IF ECH IS SET AND,
9572                                    ;       DATA FIELD ERRORS ARE NOT ENABLED, OR
9573                                    ;       ECI IS SET, OR
9574                                    ;       DCK IS NOT SET.
9575  043306  012746  177777            MOV     #-1,-(SP)       ;ASSUME ENABLED
9576  043312  032737  000100  044472    BIT     #ECH,5OO$       ;ARE ERRORS ENABLED ??
9577  043320  001410                    BEQ     265$            ;NO !!
9578  043322  032737  004000  001360    BIT     #ECI,RMOFI      ;IS ECI SET ??
9579  043330  001004                    BNE     265$            ;YES !!
9580  043332  032737  100000  001342    BIT     #DCK,RMER1I     ;IS DCK ALSO SET ??
9581  043340  001002                    BNE     270$            ;YES !!
9582  043342  042716  000100          265$:   BIC     #ECH,(SP)       ;DISABLE ECH
9583  043346  013737  001342  001142  270$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
9584  043354  042637  001142            BIC     (SP)+,$BDDAT    ;CLEAR ECH IF ENABLED
9585  043360  001412                    BEQ     275$            ;BRANCH IF ECH IS OK
9586  043362  062716  000004            ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
9587  043366  112776  000031  000000    MOVB    #31,@(SP)       ;WRITE ERROR NUMBER IN CALL
9588  043374  162716  000002            SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
9589  043400  004736                    JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
9590  043402  162716  000010            SUB     #10,(SP)        ;MOVE SP TO NO ERROR
9591  043406                          275$:
9592
9593                                    ;;*****************************************************************
9594                                    ;PERFORM THE REMAINING ERROR CHECKS ONLY FOR DATA TRANSFER COMMANDS
9595
```

```
 9596  043406  022737  000030  044500          CMP     #SEARCH,515$    ;WAS DATA TRANSFERRED??
 9597  043414  103402                          BLO     280$            ;YES!!
 9598  043416  000137  044444                  JMP     355$
 9599
```

```
                                        ;THE FOLLOWING STATUS CHECKS ARE FOR DATA TRANSFER COMMANDS ONLY
                                        ;REPORT ERROR IF RMWC NOT ZERO AND TRE IS ZERO
043422  013737  001330  001142  280$:   MOV     RMWCI,$BDDAT    ;WORD COUNT ZERO??
043430  001421                          BEQ     285$            ;YES
043432  032737  040000  001326          BIT     #TRE,RMCS1I     ;TRANSFER ERROR DETECTED??
043440  001015                          BNE     285$            ;YES!!
043442  062716  000004                  ADD     #4,(SP)
043446  112776  000015  000000          MOVB    #15.,@(SP)      ;ERROR NUMBER
043454  005037  001140                  CLR     $GDDAT          ;GOOD DADA FOR TYPEOUT
043460  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
043464  004736                          JSR     PC,@(SP)+       ;REPORT WORD COUNT NOT ZERO
043466  162716  000010                  SUB     #10,(SP)        ;RESTORE (SP)
043472  000240                          NOP
                                        ;REPORT ERROR IF RMBA IS NOT CORRECT
043474  013737  001330  001140  285$:   MOV     RMWCI,$GDDAT    ;NUMBER OF WORDS TRANSFERRED
043502  163737  001400  001140          SUB     RMWCO,$GDDAT
043510  006337  001140                  ASL     $GDDAT
043514  063737  001402  001140          ADD     RMBAO,$GDDAT    ;EXPECTED BUS ADDRESS
043522  032737  000010  001336          BIT     #BAI,RMCS2I     ;WAS BAI SET ??
043530  001403                          BEQ     290$            ;NO !!
043532  013737  001402  001140          MOV     RMBAO,$GDDAT    ;ADDRESS SHOULD NOT HAVE CHANGED
043540  023737  001140  001332  290$:   CMP     $GDDAT,RMBAI    ;BUS ADDRESS OK??
043546  001416                          BEQ     295$            ;YES!!
043550  013737  001332  001142          MOV     RMBAI,$BDDAT    ;BAD DATA FOR TYPEOUT
043556  062716  000004                  ADD     #4,(SP)
043562  112776  000016  000000          MOVB    #16.,@(SP)      ;ERROR NUMBER
043570  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
043574  004736                          JSR     PC,@(SP)+       ;REPORT UNEXPECTED ADDRESS
043576  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
043602  000240                          NOP
                                        ;COMPUTE NUMBER OF SECTORS TRANSFERRED
043604  005046                  295$:   CLR     -(SP)           ;NUMBER OF SECTORS TRANSFERRED
043606  013746  001330                  MOV     RMWCI,-(SP)     ;NUMBER OF WORDS TRANSFERRED
043612  163716  001400                  SUB     RMWCO,(SP)
043616  012746  000400                  MOV     #256.,-(SP)     ;NUMBER OF WORDS PER SECTOR
043622  032737  000002  001376          BIT     #BIT1,RMCS1O    ;HEADER & DATA COMMAND ??
043630  001402                          BEQ     300$            ;NO !!
043632  012716  000402                  MOV     #258.,(SP)      ;YES - CHANGE WORDS PER SECTOR
043636  005266  000004          300$:   INC     4(SP)           ;INCREMENT SECTOR COUNT
043642  161666  000002                  SUB     (SP),2(SP)      ;SUBTRACT ONE SECTOR'S WORTH
043646  003373                          BGT     300$            ;CONTINUE IF NOT DONE
043650  022626                          CMP     (SP)+,(SP)+     ;STRIP 2 FROM STACK
                                        ;COMPUTE EXPECTED SECTOR, TRACK AND CYLINDER ADDRESS FROM NUMBER OF SECTORS
043652  013737  001404  044476          MOV     RMDAO,510$      ;STORE ORIGINAL SECTOR
043660  013737  001404  044474          MOV     RMDAO,505$      ;STORE ORIGINAL TRACK
043666  013737  001432  044472          MOV     RMDCO,500$      ;STORE ORIGINAL CYLINDER
043674  042737  177740  044476          BIC     #^C37,510$
043702  000337  044474                  SWAB    505$
043706  042737  177770  044474          BIC     #^C7,505$
043714  062637  044476                  ADD     (SP)+,510$

043720  023727  044476  000040  305$:   CMP     510$,#32.       ;SECTOR OVEFLOWED??
043726  103420                          BLO     315$            ;NO!!
043730  023727  044476  000240          CMP     510$,#(5*32.)   ;CYLINDERS WORTH??
043736  103406                          BLO     310$            ;NO!!
043740  005237  044472                  INC     500$            ;YES INCREMENT CYLINDER
```

```
                                        SUB     #<5*32.>,510$   ;ADJUST SECTOR
                                        BR      315$            ;TRY AGAIN
                                310$:   INC     510$            ;INCREMENT TRACK
                                        SUB     #32.,510$       ;ADJUST SECTOR
                                        BR      305$            ;TRY AGAIN

                                315$:   CMP     505$,#5 ;TRACK OVERFLOWED??
                                        BLO     320$            ;NO!!
                                        INC     500$            ;INCREMENT CYLINDER
                                        SUB     #5,505$ ;ADJUST TRACK
                                        BR      315$            ;TRY AGAIN
                                ;REPORT ERROR IF "LBT" IS NOT CORRECT
                                320$:   CLR     $GDDAT          ;SET GOOD DATA FOR LBT=0
                                        CMP     #823.,500$      ;SHOULD LBT BE SET??
                                        BHI     325$            ;NO!!
                                        BIT     #IAE,RMER1I     ;WAS IAE SET ??
                                        BNE     325$            ;YES - LBT SHOULD NOT BE SET
                                        MOV     #LBT,$GDDAT     ;SET GOOD DATA FOR LBT=1
                                325$:   MOV     RMDS1,$BDDAT    ;BAD DATA FOR TYPEOUT
                                        BIC     #^CLBT,$BDDAT
                                        CMP     $GDDAT,$BDDAT   ;IS LBT CORRECT??
                                        BEQ     330$            ;YES!!
                                        ADD     #4,(SP)
                                        MOVB    #17,@(SP)       ;ERROR NUMBER
                                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                        JSR     PC,@(SP)+       ;REPORT LBT IS WRONG
                                        SUB     #10,(SP)                ;RESTORE (SP)
                                        NOP
                                ;REPORT ERROR IF "AOE" IS INCORRECT
                                330$:   CLR     $GDDAT          ;SET FOR AOE=0
                                        BIT     #IAE,RMER1I     ;WAS "IAE" DETECTED??
                                        BNE     340$            ;YES-"AOE" SHOULD BE ZERO
                                        CMP     #823.,500$      ;SHOULD AOE BE SET??
                                        BHI     340$            ;NO!!
                                        TST     505$            ;MAYBE
                                        BNE     335$            ;YES
                                        TST     510$
                                        BNE     335$            ;YES !!
                                        BIT     #F2,515$        ;WAS THIS READ OR WRITE CHECK ??
                                        BEQ     340$            ;NO !!
                                        TST     RMWCI           ;WAS ALL DATA TRANSFERRED ??
                                        BEQ     340$            ;YES !!
                                335$:   MOV     #AOE,$GDDAT     ;SET FOR AOE=1
                                        CLR     505$            ;CLEAR EXPECTED TRACK
                                        MOV     #1,510$ ;EXPECT SECTOR=1
                                340$:   MOV     RMER1I,$BDDAT   ;BAD DATA FOR TYPEOUT
                                        BIC     #^CAOE,$BDDAT
                                        CMP     $GDDAT,$BDDAT   ;IS AOE CORRECTY??
                                        BEQ     345$            ;YES!!
                                        ADD     #4,(SP)
                                        MOVB    #20,@(SP)       ;ERROR NUMBER
                                        SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
                                        JSR     PC,@(SP)+       ;REPORT AOE IS WRONG
                                        SUB     #10,(SP)                ;RESTORE (SP)
                                        NOP
                                ;REPORT ERROR IF RMDA IS NOT CORRECT
```

```
 9712  044270  032737  002000  001342  345$:   BIT     #IAE,RMERII     ;WAS THERE AN IAE ERROR ??
 9713  044276  001062                          BNE     355$            ;YES - DONT CHECK RMDA,RMDC
 9714  044300  013737  044474  001140          MOV     505$,$GDDAT     ;SETUP EXPECTED DISK ADDRESS
 9715  044306  000337  001140                  SWAB    $GDDAT
 9716  044312  113737  044476  001140          MOVB    510$,$GDDAT
 9717  044320  013737  001334  001142          MOV     RMDAI,$BDDAT    ;SETUP RECEIVED DISK ADDRESS
 9718  044326  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
 9719  044334  001413                          BEQ     350$            ;BRANCH IF EQUAL
 9720  044336  062716  000004                  ADD     #4,(SP)
 9721  044342  112776  000021  000000          MOVB    #21,2(SP)       ;ERROR NUMBER
 9722  044350  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9723  044354  004736                          JSR     PC,@(SP)+       ;REPORT BAD DISK ADDRESS
 9724  044356  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
 9725  044362  000240                          NOP
 9726                          ;REPORT ERROR IF RMDC IS INCORRECT
 9727  044364  013737  044472  001140  350$:   MOV     500$,$GDDAT     ;SETUP EXPECTED CYLINDER
 9728  044372  042737  176000  001140          BIC     #^C1777,$GDDAT
 9729  044400  013737  001362  001142          MOV     RMDCI,$BDDAT    ;SETUP RECEIVED CYLINDER
 9730  044406  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE CYLINDERS
 9731  044414  001413                          BEQ     355$            ;BRANCH IF EQUAL
 9732  044416  062716  000004                  ADD     #4,(SP)
 9733  044422  112776  000022  000000          MOVB    #22,2(SP)       ;ERROR NUMBER
 9734  044430  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9735  044434  004736                          JSR     PC,@(SP)+       ;REPORT BAD CYLINDER
 9736  044436  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
 9737  044442  000240                          NOP
```

```
9738  044444  062716  000004          355$:   ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
9739  044450  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
9740  044454  001403                          BEQ     360$
9741  044456  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR RETURN
9742  044462  000402                          BR      365$
9743  044464  162716  000004          360$:   SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
9744  044470  000207                  365$:   RTS     PC
9745
9746  044472  000000                  500$:   .WORD   0               ;CYLINDER
9747  044474  000000                  505$:   .WORD   0               ;TRACK
9748  044476  000000                  510$:   .WORD   0               ;SECTOR
9749  044500  000000                  515$:   .WORD   0               ;FUNCTION CODE
9750
9751
```

```
                                .SBTTL  DEVICE SELECT SUBROUTINE

                                ; THIS SUBROUTINE SELECTS THE DEVICE, GETTING THE DEVICE NUMBER FROM THE
                                ; TEST QUEUE.

                                ;CALL:
                                ;(1)    JSR     PC,DEVSEL
                                ;(2)    BR      ??              RETURN IF NO ERROR
                                ;(3)    NOP                     RETURN IF ERROR
                                ;(4)    ERROR                   ERROR DEFINED BY SUBROUTINE

044502                          DEVSEL:

                                ;CLEAR USER'S ERROR CALL
044502  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
044506  105076  000000                  CLRB    @(SP)           ;CLEAR LOW ORDER BYTE OF CALL
044512  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK
                                ;SAVE USER'S INFORMATION AND SETUP REGISTERS
044516  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
044522  013746  000006                  MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
044526  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
044530  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
044532  012737  044652  000004          MOV     #20$,ERRVEC     ;SETUP FOR BUS TIMEOUT
044540  012737  000300  000006          MOV     #PR6,ERRVEC+2
044546  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
044552  013701  001446                  MOV     TSTQUE,R1       ;R1 POINTS TO DEVICE NUMBER

                                ;SELECT DEVICE AND VERIFY THAT DEVICE IS AVAILABLE
044556  111160  000010                  MOVB    (R1),RMCS2(R0)  ;WRITE UNIT SELECT BITS
044562  016037  000000  001176          MOV     RMCS1(R0),$TMP1 ;GET "DVA" STATUS
044570  016037  000010  001174          MOV     RMCS2(R0),$TMP0 ;GET "NED" STATUS
044576  032737  010000  001174          BIT     #NED,$TMP0      ;IS DEVICE NONEXISTENT??
044604  001407                          BEQ     10$             ;NO!!
044606  062766  000004  000010          ADD     #4,10(SP)       ;MOVE SP TO USERS ERROR CALL
044614  112776  000111  000010          MOVB    #111,@10(SP)    ;WRITE ERROR NUMBER
044622  000422                          BR      30$             ;
044624  032737  004000  001176  10$:    BIT     #DVA,$TMP1      ;IS DEVICE AVAILABLE??
044632  001021                          BNE     35$             ;YES!!
044634  062766  000004  000010          ADD     #4,10(SP)
044642  112776  000112  000010          MOVB    #112,@10(SP)
044650  000407                          BR      30$

044652                          20$:

                                ;HANDLE BUS TIMEOUT
044652  022626                          CMP     (SP)+,(SP)+     ;ADJUST SP
044654  062766  000004  000010          ADD     #4,10(SP)
044662  112776  000113  000010          MOVB    #113,@10(SP)
044670  162766  000002  000010  30$:    SUB     #2,10(SP)       ;MOVE SP TO RETURN IF ERROR

044676                          35$:

                                ;RESTORE USERS DATA AND RETURN TO ADDRESS ON STACK
044676  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
044700  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
```

```
 9808  044702  012637  000006                MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
 9809  044706  012637  000004                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
 9810  044712  000207                        RTS     PC              ;EXIT
```

```
  9811                                  .SBTTL  SEEK STATUS CHECK SUBROUTINE
  9812                                  ;
  9813                                  ;THIS SUBROUTINE VERIFIES THE RESULTS OF SEEK TESTS USING STATUS
  9814                                  ;STORED IN THE GET BUFFER AND TEST PARAMETERS STORED IN THE PUT BUFFER.
  9815                                  ;
  9816                                  ;
  9817                                  ;THE SUBROUTINE RETURNS TO THE CALLING ROUTINE IF AN ERROR IS DETECTED
  9818                                  ;AFTER HAVING LOADED THE APPROPRIATE ERROR NUMBER IN THE "ERROR" TRAP
  9819                                  ;OF THE CALLING ROUTINE.  SEEK STATUS IS CHECKED AS FOLLOWS:
  9820                                  ;
  9821                                  ;CALL:
  9822                                  ;(1)    JSR     PC,SEKSTS
  9823                                  ;       BR      ???             RETURN HERE IF NO ERROR
  9824                                  ;       NOP                     RETURN HERE TO REPORT AN ERROR
  9825                                  ;       ERROR                   ERROR NUMBER DEFINED BY SUB
  9826                                  ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
  9827                                  ;       ???                     RETURN HERE IF NO MORE ERRORS
  9828
  9829  044714                          SEKSTS:
  9830
  9831                                  ;CLEAR USER'S ERROR  CALL
  9832  044714  000240                          NOP
  9833  044716  062716  000004                 ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
  9834  044722  105076  000000                 CLRB    @(SP)           ;CLEAR ERROR NUMBER
  9835  044726  162716  000004                 SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
  9836  044732  005037  046152                 CLR     300$            ;CLEAR ERROR FLAGS
  9837
  9838                                  ;TEST FOR MASSBUS CONTROL BUS PARITY ERROR WHEN WRITING
  9839                                  ;LOCAL REGISTERS, I.E., "PAR"  = 1 AND "DPE" = 0
  9840  044736  032737  000010  001342          BIT     #PAR,RMER1I     ;WAS PARITY ERROR DETECTED??
  9841  044744  001424                          BEQ     1$              ;NO!!
  9842  044746  032737  000010  001370         BI      #DPE,RMER2I     ;WAS IT DUE TO CONTROL BUS??
  9843  044754  001020                          BNE     1$              ;NOT SURE!!
  9844
  9845                                  ;REPORT REGISTER PARITY ERROR VIA USER'S ERROR CALL
  9846  044756  005037  001140                 CLR     $GDDAT          ;EXPECTED STATUS
  9847  044762  013737  001342  001142         MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
  9848  044770  062716  000004                 ADD     #4,(SP)         ;MOVE STACK TO USER'S ERROR
  9849  044774  112776  000050  000000         MOVB    #50,@(SP)       ;ERROR #50
  9850  045002  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
  9851  045006  004736                          JSR     PC,@(SP)+
  9852  045010  162716  000010                 SUB     #10,(SP)        ;RESTORE STACK
  9853  045014  000437                          BR      3$              ;IAE SHOULD  BE ZERO
  9854
  9855
  9856                                  ;DETERMINE THE VALUE OF "IAE" STATUS BASED ON TRACK,SECTOR AND
  9857                                  ;CYLINDER ADDRESS USED DURING SEEK OPERATION
  9858  045016  012737  002000  001140  1$:     MOV     #IAE,$GDDAT     ;SET UP FOR IAE = 1
  9859  045024  052737  040000  046152          BIS     #SKI,300$       ;SET SKI ERROR FLAG
  9860  045032  023727  001432  001466         CMP     RMDC0,#822.     ;CYLINDER > 822??
  9861  045040  101025                          BHI     3$              ;YES!!
  9862  045042  042737  040000  046152         BIC     #SKI,300$       ;CLEAR SKI ERROR FLAG
  9863  045050  123727  001404  000037         CMPB    RMDA0,#31.      ;SECTOR > 31??
  9864  045056  101016                          BHI     3$              ;YES!!
  9865  045060  123727  001404  000035         CMPB    RMDA0,#29.      ;SECTOR > 29 ??
  9866  045066  101404                          BLOS    2$              ;NO!!
```

```
 9867  045070  032737  010000  001430          BIT     #FMT16,RMOFO    ;30 SECTOR FORMAT??
 9868  045076  001406                          BEQ     3$              ;YES!!
 9869  045100  123727  001405  000004  2$:     CMPB    RMDA0+1,#4.     ;TRACK  >4??
 9870  045106  101002                          BHI     3$              ;YES!!
 9871  045110  005037  001140                  CLR     $GDDAT          ;"IAE" SHOULD BE 0
 9872
 9873                                          ;COMPARE EXPECTED AND RECEIVED "IAE" STATUS
 9874  045114  013737  001342  001142  3$:     MOV     RMER1I,$BDDAT   ;IS IAE OK??
 9875  045122  042737  175777  001142          BIC     #^CIAE,$BDDAT
 9876  045130  023737  001140  001142          CMP     $GDDAT,$BDDAT
 9877  045136  001004                          BNE     35$             ;IAE IN ERROR
 9878  045140  042737  040000  046152          BIC     #SKI,300$       ;CLEAR SKI FLAG
 9879  045146  000413                          BR      5$              ;GO CHECK NEXT ERROR
 9880  045150                                  35$:
 9881                                          ;REPORT INCORRECT "IAE" STATUS VIA USER'S ERROR CALL
 9882  045150  062716  000004                  ADD     #4,(SP)
 9883  045154  112776  000051  000000          MOVB    #51,@(SP)       ;ERROR 51
 9884  045162  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9885  045166  004736                          JSR     PC,@(SP)+       ;REPORT INCORRECT IAE
 9886  045170  162716  000010                  SUB     #10,(SP)                ;RESTORE (SP)
 9887  045174  000240                          NOP
 9888  045176                                  5$:
 9889
 9890                                          ;REPORT ANY IVC ERROR AS
 9891                                          ;       IVC ERROR WITH VOLUME VALID ZERO
 9892                                          ;       ERRONEOUS IVC ERROR, VOLUME VALID IS SET
 9893  045176  032737  010000  001370          BIT     #IVC,RMER2I     ;IVC ERROR??
 9894  045204  001427                          BEQ     52$             ;NO!!
 9895  045206  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
 9896  045212  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
 9897  045220  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
 9898  045224  112776  000060  000000          MOVB    #60,@(SP)       ;ERROR 60 IF VV = 0
 9899  045232  032737  000100  001340          BIT     #VV,RMDSI
 9900  045240  001403                          BEQ     51$
 9901  045242  112776  000061  000000          MOVB    #61,@(SP)       ;ERROR 61 IF VV  = 1
 9902  045250  162716  000002          51$:    SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9903  045254  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
 9904  045256  162716  000010                  SUB     #10,(SP)        ;RESTORE SP
 9905  045262  000240                          NOP
 9906
 9907  045264  013737  001370  001142  52$:     MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
 9908  045272  042737  137777  001142          BIC     #^CSKI,$BDDAT   ;CLEAR DONT CARES
 9909  045300  013737  046152  001140          MOV     300$,$GDDAT     ;GET EXPECTED SKI STATUS
 9910  045306  042737  137777  001140          BIC     #^CSKI,$GDDAT   ;CLEAR DONT CARES
 9911  045314  001417                          BEQ     53$             ;BRANCH IF 0 EXPECTED
 9912
 9913                                          ;REPORT ERROR IF SKI IS NOT SET (IAE WAS NOT DETECTED)
 9914  045316  032737  040000  001142          BIT     #SKI,$BDDAT     ;WAS SKI DETECTED ??
 9915  045324  001032                          BNE     54$             ;YES !!
 9916  045326  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 9917  045332  112776  000267  000000          MOVB    #267,@(SP)      ;WRITE ERROR NUMBER
 9918  045340  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO ERROR RETURN
 9919  045344  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
 9920  045346  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
 9921  045352  000443                          BR      6$              ;GO TO NEXT ERROR CHECK
 9922  045354                                  53$:
```

```
                                        ;REPORT ERROR IF SKI IS SET
045354  032737  040000  001142          BIT     #SKI,$BDDAT     ;IS SKI SET ??
045362  001413                          BEQ     54$             ;NO - SKI IS OK
045364  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR
045370  112776  000054  000000          MOVB    #54,@(SP)       ;LOAD ERROR NUMBER
045376  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
045402  004736                          JSR     PC,@(SP)+       ;REPORT SEEK ERROR
045404  162716  000010                  SUB     #10,(SP)        ;RESTORE (SP)
045410  000240                          NOP

                                        ;REPORT ANY DEVICE CHECK
045412  032737  000200  001370  54$:    BIT     #DVC,RMER2I     ;WAS THERE DVC DURING SEEK??
045420  001420                          BEQ     6$              ;NO!!
045422  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS
045426  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
045434  062716  000004                  ADD     #4,(SP)
045440  112776  000055  000000          MOVB    #55,@(SP)       ;ERROR  #55
045446  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
045452  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
045454  162716  000010                  SUB     #10,(SP)        ;RESTORE SP
045460  000240                          NOP

                                        ;REPORT ANY "OPI" ERROR AS OPI WITH MOL  =0, OR  OPI
                                        ;BECAUSE ON CYLINDER LATCH DIDN'T RESET
045462  032737  020000  001342  6$:     BIT     #OPI,RMER1I     ;"OPI" ERROR??
045470  001427                          BEQ     8$              ;NO!!
045472  005037  0011                    CLR     $GDDAT          ;EXPECTED STATUS
045476  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
045504  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR
045510  112776  000052  000000          MOVB    #52,@(SP)       ;LOAD ERROR NUMBER
045516  032737  010000  001340          BIT     #MOL,RMDSI      ;WAS MEDIUM ON LINE??
045524  001403                          BEQ     7$              ;NO!!
045526  112776  000053  000000          MOVB    #53,@(SP)       ;YES - CHANGE ERROR NUMBER
045534  162716  000002          7$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
045540  004736                          JSR     PC,@(SP)+       ;REPORT "OPI" ERROR
045542  162716  000010                  SUB     #10,(SP)        ;RESTORE (SP)
045546  000240                          NOP

                                        ;SEE IF "PIP" IS 0, AND "ATA","MOL"AND"VV" =1
045550  013746  001340          8$:     MOV     RMDSI,-(SP)
045554  042716  047677                  BIC     #^C<ATA!PIP!MOL!VV>,(SP)
045560  022726  110100                  CMP     #ATA!MOL!VV,(SP)+
045564  001002                          BNE     9$              ;ERROR IN RMDS
045566  000137  046122                  JMP     14$             ;RMDS IS OK

                                        ;REPORT ERROR IF MOL = 0 AND OFI = 0
045572  032737  010000  001340  9$:     BIT     #MOL,RMDSI      ;IS MOL RESET??
045600  001030                          BNE     10$             ;NO - MOL IS SET
045602  032737  020000  001342          BIT     #OPI,RMER1I     ;WAS OPI SET
045610  001024                          BNE     10$             ;YES - DONT REPORT ERROR
045612  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
045620  052737  010000  001140          BIS     #MOL,$GDDAT
045626  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
045634  062716  000004                  ADD     #4,(SP)
045640  112776  000062  000000          MOVB    #62,@(SP)
```

```
 9979  045646  162716  000002            SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9980  045652  004736                    JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
 9981  045654  162716  000010            SUB     #10,(SP)
 9982  045660  000240                    NOP
 9983
 9984                                  ;REPORT AN ERROR IF "PIP" IS STIL SET AND SKI NOT SET
 9985  045662  032737  020000  001340  10$:     BIT     #PIP,RMDSI      ;IS "PIP" STILL SET??
 9986  045670  001430                    BEQ     11$             ;NO!!
 9987  045672  032737  040000  001370          BIT     #SKI,RMER2I     ;WAS "SKI"SET??
 9988  045700  001024                    BNE     11$             ;YES-DONT REPORT PIP
 9989  045702  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
 9990  045710  042737  020000  001142          BIC     #PIP,$BDDAT
 9991  045716  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
 9992  045724  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR
 9993  045730  112776  000056  000000          MOVB    #56,@(SP)       ;LOAD ERROR NUMBER
 9994  045736  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
 9995  045742  004736                    JSR     PC,@(SP)+       ;REPORT "PIP" SET AFTER SEEK
 9996  045744  162716  000010            SUB     #10,(SP)                ;RESTORE (SP)
 9997  045750  000240                    NOP
 9998
 9999                                  ;REPORT AN ERROR IF "ATA" IS NOT SET
10000  045752  032737  100000  001340  11$:     BIT     #ATA,RMDSI      ;WAS "ATA" SET ??
10001  045760  001024                    BNE     13$             ;YES!!
10002  045762  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10003  045770  052737  110600  001140          BIS     #ATA!MOL!DPR!DRY,$GDDAT
10004  045776  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10005  046004  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR
10006  046010  112776  000057  000000          MOVB    #57,@(SP)       ;LOAD ERROR NUMBER
10007  046016  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10008  046022  004736                    JSR     PC,@(SP)+       ;REPORT ATTENTION NOT SET DURING
10009                                                          ;SEEK TEST
10010  046024  162716  000010            SUB     #10,(SP)                ;RESTORE (SP)
10011  046030  000240                    NOP
10012
10013                                  ;REPORT ERROR IF VOLUME VALID IS RESET AND IVC IS ZERO
10014  046032  032737  000100  001340  13$:     BIT     #VV,RMDSI       ;IS VV = 0 ??
10015  046040  001030                    BNE     14$             ;NO!!
10016  046042  032737  010000  001370          BIT     #IVC,RMER2I     ;IS IVC ALSO 0 ??
10017  046050  001024                    BNE     14$             ;NO - IVC IS SET
10018  046052  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10019  046060  052737  000100  001140          BIS     #VV,$GDDAT
10020  046066  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10021  046074  062716  000004                  ADD     #4,(SP)
10022  046100  112776  000064  000000          MOVB    #64,@(SP)       ;ERROR #64
10023  046106  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10024  046112  004736                    JSR     PC,@(SP)+
10025  046114  162716  000010            SUB     #10,(SP)
10026  046120  000240                    NOP
10027  046122                          14$:
10028
10029                                  ;MODIFY THE RETURN ADDRESS IF AN ERROR WAS DETECTED
10030  046122  000240                    NOP
10031  046124  062716  000004                  ADD     #4,(SP)         ;MOVE (SP) TO ERROR CALL
10032  046130  105776  000000                  TSTB    @(SP)           ;WAS ERROR CALLED??
10033  046134  001403                    BEQ     15$             ;NO!!
10034  046136  062716  000004                  ADD     #4,(SP)         ;MOVE TO ERROR RETURN
```

```
10035  046142  000402                    BR      16$
10036
10037  046144  162716  000004    15$:    SUB     #4,(SP)         ;MOVE (SP) TO NO ERROR RETURN
10038  046150  000207            16$:    RTS     PC              ;RETURN
10039
10040  046152  000000            300$:   .WORD   0               ;ERROR FLAGS
10041
```

```
10042                                           .SBTTL  CONTROLLER CLEAR SUBROUTINE
10043
10044                                           ;THIS SUBROUTINE CLEARS THE MASSBUS CONTROLLER, MASSBUS ADAPTERS,
10045                                           ;AND DRIVES, THEN SELECTS THE DRIVE.
10046                                           ;
10047                                           ;CALL:  JSR     PC,CNTCLR
10048                                           ;       BR      ???             RETURN HERE IF NO ERROR FOUND
10049                                           ;       NOP                     RETURN HERE IF ANY ERROR FOUND
10050                                           ;       ERROR                   SUB DEFINES ERROR NUMBER
10051                                           ;       ???
10052
10053   046154                                  CNTCLR:
10054   046154  010046                                 MOV     RO,-(SP)        ;;PUSH RO ON STACK
10055   046156  010146                                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10056   046160  013746  000004                         MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
10057   046164  013746  000006                         MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
10058   046170  012737  046230  000004                 MOV     #10$,ERRVEC     ;SETUP FOR BUS TIMEOUT
10059   046176  012737  000300  000006                 MOV     #PR6,ERRVEC+2
10060   046204  013700  001276                         MOV     $BASE,RO        ;RO=UNIBUS ADDRESS
10061   046210  012760  000040  000010                 MOV     #CLR,RMCS2(RO)  ;CLEAR MASSBUS
10062   046216  013701  001446                         MOV     TSTQUE,R1
10063   046222  111160  000010                         MOVB    (R1),RMCS2(RO)  ;SELECT DEVICE
10064   046226  000412                                 BR      20$
10065   046230  022626                          10$:   CMP     (SP)+,(SP)+     ;ADJUST STACK
10066   046232  062766  000004  000010                 ADD     #4,10(SP)       ;MOVE SP TO USER'S ERROR CALL
10067   046240  112776  000007  000010                 MOVB    #7,@10(SP)      ;WRITE THE ERROR NUMBER
10068   046246  162766  000002  000010                 SUB     #2,10(SP)
10069   046254                                  20$:
10070   046254  012637  000006                         MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
10071   046260  012637  000004                         MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
10072   046264  012601                                 MOV     (SP)+,R1        ;;POP STACK INTO R1
10073   046266  012600                                 MOV     (SP)+,RO        ;;POP STACK INTO RO
10074   046270  000207                                 RTS     PC
10075
```

```
10076                                   .SBTTL  CONTROLLER CLEAR STATUS CHECK SUBROUTINE
10077
10078                                   ;THIS SUBROUTINE VERIFIES THAT THE RM03 SUBSYSTEM IS INITIALIZED BASED ON
10079                                   ;STATUS STORED IN THE GET BUFFER.  THIS SUBROUTINE SHOULD ONLY BE
10080                                   ;USED FOLLOWING A CONTROLLER CLEAR OPERATION, I.E., WRITING A 1 IN BIT
10081                                   ;5 OF RMCS2, BECAUSE THE ERROR MESSAGES ARE BASED ON THAT CONDITION.
10082
10083                                   ;STATUS PERTINENT TO THE DEVICE IS NOT CHECKED.  IN PARTICULAR, THE
10084                                   ;FOLLOWING STATUS BITS ARE NOT CHECKED:
10085                                   ;
10086                                   ;       ATA,ERR,PIP,MOL,WRL,LBT,PGM,VV,OM,UNS,SKI,DVC
10087                                   ;
10088
10089                                   ;CALL:
10090                                   ;(1)    JSR     PC,CLRSTS
10091                                   ;       BR      ???             RETURN HERE IF NO ERROR
10092                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
10093                                   ;       ERRCR                   ERROR NUMBER DEFINED BY SUB
10094                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10095                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10096
10097   046272                          CLRSTS:
10098
10099                                   ;CLEAR USER'S ERROR CALL
10100   046272  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO ERROR
10101   046276  105076  000000                 CLRB    @(SP)           ;CLEAR ERROR NUMBER
10102   046302  162716  000004                 SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR
10103                                   ;REPORT ERROR IF RMCS1 NOT INITIALIZED
10104   046306  013737  001326  001142 4$:     MOV     RMCS1I,$BDDAT   ;VERIFY RMCS1
10105   046314  042737  100000  001142         BIC     #SC,$BDDAT      ;IGNORE SPECIAL CONDITION
10106   046322  012737  004200  001140         MOV     #DVA!RDY,$GDDAT ;EXPECT DVA & RDY
10107   046330  023737  001140  001142         CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED, RECEIVED
10108   046336  001413                 BEQ     5$              ;BRANCH IF EQUAL
10109   046340  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10110   046344  112776  000126  000000         MOVB    #126,@(SP)      ;WITE ERROR NUMBER IN CALL
10111   046352  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10112   046356  004736                 JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10113   046360  162716  000010                 SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10114   046364  000240                 NOP
10115                                   ;REPORT ERROR IF RMBA NOT RESET
10116   046366  005037  001140 5$:     CLR     $GDDAT          ;VERIFY RMBA IS ZERO
10117   046372  013737  001332  001142         MOV     RMBAI,$BDDAT
10118   046400  001413                 BEQ     7$              ;BRANCH IF ZERO
10119   046402  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10120   046406  112776  000127  000000         MOVB    #127,@(SP)      ;WITE ERROR NUMBER IN CALL
10121   046414  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10122   046420  004736                 JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10123   046422  162716  000010                 SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10124   046426  000240                 NOP
10125                                   ;REPORT ERROR IF RMCS2 NOT INITIALIZED
10126   046430  013737  001336  001142 7$:     MOV     RMCS2I,$BDDAT   ;VERIFY RMCS2
10127   046436  010146                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10128   046440  005046                 CLR     -(SP)           ;EXPECT IR & UNIT NUMBER
10129   046442  013701  001446                 MOV     TSTQUE,R1       ;R1 = ADDRESS OF TEST QUE
10130   046446  111116                 MOVB    (R1),(SP)
10131   046450  052716  000100                 BIS     #IR,(SP)
```

```
10132  046454  012637  001140          MOV     (SP)+,$GDDAT
10133  046460  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
10134  046462  023737  001140  001142  CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10135  046470  001413                  BEQ     9$              ;BRANCH IF EQUAL
10136  046472  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10137  046476  112776  000130  000000  MOVB    #130,@(SP)      ;WITE ERROR NUMBER IN CALL
10138  046504  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10139  046510  004736                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10140  046512  162716  000010          SUB     #10 (SP)        ;MOVE SP BACK TO NO ERROR
10141  046516  000240                  NOP
10142                          ;REPORT ERROR IF RMER1 NOT RESET-IGNORE UNS
10143  046520  005037  001140  9$:     CLR     $GDDAT          ;VERIFY RMER1
10144  046524  013737  001342  001142  MOV     RMER1I,$BDDAT
10145  046532  042737  040000  001142  BIC     #UNS,$BDDAT     ;IGNORE UNSAFE
10146  046540  001413                  BEQ     13$             ;BRANCH IF ZERO
10147  046542  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10148  046546  112776  000131  000000  MOVB    #131,@(SP)      ;WITE ERROR NUMBER IN CALL
10149  046554  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10150  046560  004736                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10151  046562  162716  000010          SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10152  046566  000240                  NOP
10153                          ;REPORT ERROR IF RMMR1 NOT INITIALIZED-IGNORE WC,LS,LST
10154  046570  013737  001352  001142  13$:    MOV     RMMR1I,$BDDAT   ;VERIFY RMMR
10155  046576  042737  000046  001142  BIC     #WC!LS!LST,$BDDAT       ;IGNORE WORD CLOCK, SCT, TRK
10156  046604  012737  000010  001140  MOV     #MWD,$GDDAT     ;EXPECT WRITE DATA BIT
10157  046612  023737  001140  001142  CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
10158  046620  001413                  BEQ     17$             ;BRANCH IF 0
10159  046622  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10160  046626  112776  000133  000000  MOVB    #133,@(SP)      ;WITE ERROR NUMBER IN CALL
10161  046634  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10162  046640  004736                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10163  046642  162716  000010          SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10164  046646  000240                  NOP
10165                          ;REPORT AN ERROR IF RMEC2 IS NOT RESET
10166  046650  005037  001140  17$:    CLR     $GDDAT          ;EXPECT ZEROS
10167  046654  013737  001374  001142  MOV     RMEC2I,$BDDAT   ;VERIFY RMEC2=0
10168  046662  001413                  BEQ     19$
10169  046664  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10170  046670  112776  000135  000000  MOVB    #135,@(SP)      ;WITE ERROR NUMBER IN CALL
10171  046676  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10172  046702  004736                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10173  046704  162716  000010          SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
10174  046710  000240                  NOP
10175                          ;REPORT ERROR IF RMMR2 NOT INITIALIZED-IGNORE RQA,RQB
10176  046712  013737  001366  001142  19$:    MOV     RMMR2I,$BDDAT   ;VERIFY RMMR2
10177  046720  042737  140000  001142  BIC     #RQA!RQB,$BDDAT
10178  046726  012737  011777  001140  MOV     #TST!1777,$GDDAT        ;EXPECT TEST BIT ON
10179  046734  023737  001140  001142  CMP     $GDDAT,$BDDAT
10180  046742  001413                  BEQ     21$
10181  046744  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10182  046750  112776  000136  000000  MOVB    #136,@(SP)      ;WITE ERROR NUMBER IN CALL
10183  046756  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10184  046762  004736                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10185  046764  162716  000010          SUB     #'0,(SP)        ;MOVE SP BACK TO NO ERROR
10186  046770  000240                  NOP
10187                          ;REPORT ERROR IF RMER2 NOT RESET-IGNORE SKI,DVC
```

```
10188  046772  005037  001140          21$:    CLR     $GDDAT              ;EXPECT ALL ZEROS
10189  046776  013737  001370  001142          MOV     RMER2I,$BDDAT       ;VERIFY RMER2
10190  047004  042737  040200  001142          BIC     #SKI!DVC,$BDDAT     ;IGNORE DEVICE ERRORS
10191  047012  001413                          BEQ     215$                ;BRANCH IF OTHER BITS 0
10192  047014  062716  000004                  ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
10193  047020  112776  000174  000000          MOVB    #174,@(SP)          ;WITE ERROR NUMBER IN CALL
10194  047026  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
10195  047032  004736                          JSR     PC,@(SP)+           ;REPORT ERROR VIA USER
10196  047034  162716  000010                  SUB     #10,(SP)            ;MOVE SP BACK TO NO ERROR
10197  047040  000240                          NOP
10198                                  ;REPORT ERROR IF RMDS NOT INITIALIZED
10199  047042  013737  001340  001142  215$:   MOV     RMDSI,$BDDAT        ;TEST DRIVE STATUS REGISTER
10200  047050  042737  177177  001142          BIC     #^C(DRY!DPR),$BDDAT
10201  047056  012737  000600  001140          MOV     #DPR!DRY,$GDDAT     ;EXPECTED DRIVE STATUS
10202  047064  023737  001140  001142          CMP     $GDDAT,$BDDAT       ;COMPARE EXPECTED & RECEIVED
10203  047072  001413                          BEQ     22$                 ;BRANCH IF EQUAL
10204  047074  062716  000004                  ADD     #4,(SP)             ;MOVE SP TO USER'S ERROR CALL
10205  047100  112776  000134  000000          MOVB    #134,@(SP)          ;WRITE ERROR NUMBER
10206  047106  162716  000002                  SUB     #2,(SP)             ;MOVE SP TO RETURN FOR ERROR
10207  047112  004736                          JSR     PC,@(SP)+           ;REPORT ERROR TO USER
10208  047114  162716  000010                  SUB     #10,(SP)            ;MOVE SP BACK TO NO ERROR
10209  047120  000240                          NOP
10210  047122  062716  000004          22$:    ADD     #4,(SP)             ;MOVE SP TO ERROR CALL
10211  047126  105776  000000                  TSTB    @(SP)               ;WAS AN ERROE DETECTED??
10212  047132  001403                          BEQ     23$                 ;NO!!
10213  047134  062716  000004                  ADD     #4,(SP)             ;YES - MOVE TO ERROR RETURN
10214  047140  000402                          BR      24$
10215  047142  162716  000004          23$:    SUB     #4,(SP)             ;MOVE SP TO NO ERROR RETURN
10216  047146  000240                  24$:    NOP
10217  047150  000207                          RTS     PC
```

```
10218                                   .SBTTL  PACK ACKNOWLEDGE STATUS CHECK
10219
10220                                   ;THIS SUBROUTINE CHECKS THE RESULTS OF A PACK ACKNOWLEDGE
10221                                   ;COMMAND USING THE STATUS STORED IN THE GET BUFFER.  ERRORS ARE
10222                                   ;REPORTED TO THE USER VIA THE USER'S ERROR CALL.
10223
10224                                   ;CALL:
10225                                   ;(1)    JSR     PC,ACKSTS
10226                                   ;       BR      ???             RETURN HERE IF NO ERROR
10227                                   ;       NOP                     RETURN HERE TO REPORT AN ERROR
10228                                   ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10229                                   ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10230                                   ;       ???                     RETURN HERE IF NO MORE ERRORS
10231
10232  047152                           ACKSTS:
10233
10234                                   ;CLEAR USER'S ERROR CALL
10235  047152  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10236  047156  105076  000000                  CLRB    @(SP)           ;CLEAR LOW ORDER BYTE
10237  047162  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK
10238
10239                                   ;REPORT AN ERROR  IF "VV" IS 0
10240  047166  032737  000100  001340          BIT     #VV,RMDSI       ;IS VOLUME VALID SET??
10241  047174  001024                          BNE     1$              ;YES!!
10242  047176  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10243  047204  052737  000100  001140          BIS     #VV,$GDDAT
10244  047212  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10245  047220  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10246  047224  112776  000155  000000          MOVB    #155,@(SP)      ;WRITE NUMBER IN ERROR CALL
10247  047232  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10248  047236  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR
10249  047240  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
10250  047244  000240                          NOP
10251  047246                           1$:
10252
10253                                   ;REPORT AN ERROR IF "MOL" IS 0
10254  047246  032737  010000  001340          BIT     #MOL,RMDSI      ;IS MOL SET??
10255  047254  001024                          BNE     2$              ;YES!!
10256  047256  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10257  047264  052737  010000  001140          BIS     #MOL,$GDDAT
10258  047272  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10259  047300  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10260  047304  112776  000041  000000          MOVB    #41,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10261  047312  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10262  047316  004736                          JSR     PC,@(SP)+       ;REPORT TH ERROR
10263  047320  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO BRANCH
10264  047324  000240                          NOP
10265  047326                           2$:
10266
10267                                   ;SEE IF "UNS","OPI","RMR","ILR", OR "ILF" IS SET
10268  047326  032737  060007  001342          BIT     #UNS!OPI!RMR!ILR!ILF,RMER1I
10269  047334  001570                          BEQ     7$
10270
10271                                   ;REPORT AN ERROR IF "UNS" IS SET
10272  047336  032737  040000  001342          BIT     #UNS,RMER1I     ;WAS UNS SET??
10273  047344  001424                          BEQ     3$              ;NO!!
```

```
10274  047346  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10275  047354  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10276  047362  042737  040000  001140          BIC     #UNS,$GDDAT
10277  047370  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10278  047374  112776  000042  000000          MOVB    #42,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10279  047402  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10280  047406  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10281  047410  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10282  047414  000240                          NOP
10283  047416                          3$:
10284
10285                                  ;REPORT ANY OPI ERROR
10286  047416  032737  020000  001342          BIT     #OPI,RMER1I     ;WAS OPI SET ??
10287  047424  001424                          BEQ     4$              ;NO!!
10288  047426  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10289  047434  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10290  047442  042737  020000  001140          BIC     #OPI,$GDDAT
10291  047450  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10292  047454  112776  000043  000000          MOVB    #43,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10293  047462  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10294  047466  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10295  047470  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10296  047474  000240                          NOP
10297  047476                          4$:
10298
10299                                  ;REPORT ANY RMR ERROR
10300  047476  032737  000004  001342          BIT     #RMR,RMER1I     ;WAS RMR SET??
10301  047504  001424                          BEQ     5$              ;NO!!
10302  047506  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10303  047514  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10304  047522  042737  000004  001140          BIC     #RMR,$GDDAT
10305  047530  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10306  047534  112776  000044  000000          MOVB    #44,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10307  047542  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10308  047546  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10309  047550  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10310  047554  000240                          NOP
10311  047556                          5$:
10312
10313                                  ;REPORT ANY     ILR ERROR
10314  047556  032737  000002  001342          BIT     #ILR,RMER1I     ;WAS ILR SET??
10315  047564  001424                          BEQ     6$              ;NO!!
10316  047566  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10317  047574  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10318  047602  042737  000002  001140          BIC     #ILR,$GDDAT
10319  047610  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10320  047614  112776  000045  000000          MOVB    #45,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10321  047622  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10322  047626  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10323  047630  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10324  047634  000240                          NOP
10325  047636                          6$:
10326
10327                                  ;REPORT ANY     ILF ERROR
10328  047636  032737  000001  001342          BIT     #ILF,RMER1I     ;WAS ILF SET??
10329  047644  001424                          BEQ     7$              ;NO!!
```

```
10330  047646  013737  001342  001142          MOV     RMER1I,$BODAT   ;RECEIVED STATUS
10331  047654  013737  001342  001140          MOV     RMER1I,$GODAT   ;EXPECTED STATUS
10332  047662  042737  000001  001140          BIC     #ILF,$GODAT
10333  047670  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10334  047674  112776  000046  000000          MOVB    #46,@(SP)       ;WRITE NUMBER OF ERROR IN CALL
10335  047702  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10336  047706  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10337  047710  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10338  047714  000240                          NOP
10339  047716                          7$:
10340
10341                                  ;AUGMENT RETURN ADDRESS IF ERROR WAS FOUND
10342  047716  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10343  047722  105776  000000                  TSTB    @(SP)           ;WAS ERROR FOUND??
10344  047726  001403                          BEQ     8$              ;NO!!
10345  047730  062716  000004                  ADD     #4,(SP)         ;YES - MOVE TO ERROR RETURN
10346  047734  000402                          BR      9$
10347  047736  162716  000004          8$:     SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
10348  047742  000240                  9$:     NOP
10349  047744  000207                          RTS     PC
10350
```

```
10351                                   .SBTTL  RECALIBRATE STATUS CHECK SUBROUTINE
10352
10353                                   ;THIS SUBROUTINE CHECKS THE RESULTS OF A RECALIBRATE OPERATION
10354                                   ;USING THE STATUS STORED IN THE GET BUFFER.
10355
10356                                   ;CALL:
10357                                   ;
10358                                   ;(1)    JSR     PC,RCLSTS       ;CALL SUBROUTINE
10359                                   ;       BR      ???             ;RETURN HERE IF NO ERROR
10360                                   ;       NOP                     ;RETURN HERE TO REPORT AN ERROR
10361                                   ;       ERROR                   ;ERROR NUMBER DEFINED BY SUB
10362                                   ;       JSR     PC,@(SP)+       ;GO BACK TO SUB FOR MORE ERROR CHECKS
10363                                   ;       ???                     ;RETURN HERE IF NO MORE ERRORS
10364
10365   047746                          RCLSTS:
10366
10367                                   ;CLEAR USER'S ERROR NUMBER
10368   047746  062716  000004                 ADD     #4,(SP)
10369   047752  105076  000000                 CLRB    @(SP)           ;CLEAR USER'S ERROR CALL
10370   047756  162716  000004                 SUB     #4,(SP)         ;MOVE SP BACK TO BRANCH
10371
10372
10373                                   ;SEE IF "PAR" OR "ILF" OR "OPI" OR "IAE" IS SET
10374   047762  032737  022011  001342         BIT     #OPI!PAR!ILF!IAE,RMER1I
10375   047770  001553                       BEQ     4$              ;NONE ARE SET - GO TO NEXT CHECK
10376
10377                                   ;REPORT ANY MASSBUS CONTROL BUS PARITY ERROR, I.E.,
10378                                   ;"PAR" = 1 AND "DPE" = 0
10379   047772  032737  000010  001342         BIT     #PAR,RMER1I     ;WAS "PAR" SET??
10380   050000  001430                       BEQ     1$              ;NO!!
10381   050002  032737  000010  001370         BIT     #DPE,RMER2I     ;WAS "DPE" SET??
10382   050010  001024                       BNE     1$              ;YES - NOT A REGISTER ERROR
10383   050012  013737  001342  001140         MOV RMER1I,$GDDAT       ;EXPECTED STATUS
10384   050020  042737  000010  001140         BIC     #PAR,$GDDAT
10385   050026  013737  001342  001142         MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10386   050034  062716  000004                 ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10387   050040  112776  000050  000000         MOVB    #50,@(SP)       ;WRITE ERROR NUMBER IN CALL
10388   050046  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10389   050052  004736                       JSR     PC,@(SP)+       ;GO REPORT ERROR
10390   050054  162716  000010                 SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
10391   050060  000240                       NOP
10392   050062                          1$:
10393
10394                                   ;REPORT ANY "ILF" ERROR
10395   050062  032737  000001  001342         BIT     #ILF,RMER1I     ;WAS "ILF" SET??
10396   050070  001424                       BEQ     2$              ;NO!!
10397   050072  013737  001342  001140         MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10398   050100  042737  000001  001140         BIC     #ILF,$GDDAT
10399   050106  013737  001342  001142         MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10400   050114  062716  000004                 ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10401   050120  112776  000071  000000         MOVB    #71,@(SP)       ;WRITE ERROR NUMBER IN CALL
10402   050126  162716  000002                 SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10403   050132  004736                       JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10404   050134  162716  000010                 SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
10405   050140  000240                       NOP
10406   050142                          2$:
```

```
10407
10408                                          ;REPORT ANY "OPI" ERROR AS
10409                                          ;       . OPI DUE TO "MOL" = 0
10410                                          ;       . OPI BECAUSE ON CYLINDER LATCH DIDN'T RESET
10411  050142  032737  020000  001342          BIT     #OPI,RMER1I     ;WAS OPI SET??
10412  050150  001433                          BEQ     31$             ;NO!!
10413  050152  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10414  050160  042737  020000  001140          BIC     #OPI,$GDDAT
10415  050166  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10416  050174  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10417  050200  112776  000072  000000          MOVB    #72,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
10418  050206  032737  010000  001340          BIT     #MOL,RMDSI      ;WAS "MOL" = 0??
10419  050214  001403                          BEQ     3$              ;YES!!
10420  050216  112776  000073  000000          MOVB    #73,@(SP)       ;NO - CHANGE ERROR NUMBER
10421  050224  162716  000002          3$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10422  050230  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10423  050232  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
10424  050236  000240                          NOP
10425  050240                          31$:
10426
10427                                          ;REPORT AN ERROR IF "IAE" IS SET
10428  050240  032737  002000  001342          BIT     #IAE,RMER1I     ;IS "IAE" SET??
10429  050246  001424                          BEQ     4$              ;NO!!
10430  050250  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10431  050256  042737  002000  001140          BIC     #IAE,$GDDAT
10432  050264  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10433  050272  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10434  050276  112776  000070  000000          MOVB    #70,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
10435  050304  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10436  050310  004736                          JSR     PC,@(SP)+       ;REPORT ERROR
10437  050312  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
10438  050316  000240                          NOP
10439  050320                          4$:
10440
10441                                          ;SEE IF "SKI" OR "IVC" OR "DVC" IS SET
10442  050320  032737  050200  001370          BIT     #SKI!IVC!DVC,RMER2I
10443  050326  001517                          BEQ     8$              ;NONE OF THE BITS ARE SET
10444
10445
10446                                          ;REPORT ANY "IVC" ERROR AS
10447                                          ;       .IVC WITH VV = 0
10448                                          ;       .ERRONEOUS IVC ERROR
10449  050330  032737  010000  001370          BIT     #IVC,RMER2I     ;WAS IVC SET??
10450  050336  001433                          BEQ     6$              ;NO!!
10451  050340  013737  001370  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
10452  050346  042737  010000  001140          BIC     #IVC,$GDDAT
10453  050354  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10454  050362  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10455  050366  112776  000074  000000          MOVB    #74,@(SP)       ;;WRITE ERROR NUMBER IN CALL
10456  050374  032737  000100  001340          BIT     #VV,RMDSI       ;WAS VV = 0??
10457  050402  001403                          BEQ     5$              ;YES!!
10458  050404  112776  000075  000000          MOVB    #75,@(SP)       ;NO - CHANGE ERROR NUMBER
10459  050412  162716  000002          5$:     SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10460  050416  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10461  050420  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
10462  050424  000240                          NOP
```

```
10463  050426                                  6$:
10464
10465                                          ;REPORT ANY "SKI" ERROR
10466  050426  032737  040000  001370                BIT     #SKI,RMER2I     ;WAS SKI SET??
10467  050434  001424                              BEQ     7$              ;NO!!
10468  050436  013737  001370  001140              MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
10469  050444  042737  040000  001140              BIC     #SKI,$GDDAT
10470  050452  013737  001370  001142              MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10471  050460  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10472  050464  112776  000076  000000              MOVB    #76,@(SP)       ;WRITE ERROR NUMBER
10473  050472  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10474  050476  004736                              JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10475  050500  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO BRANCH
10476  050504  000240                              NOP
10477  050506                                  7$:
10478
10479                                          ;REPORT ANY "DVC" ERROR
10480  050506  032737  000200  001370                BIT     #DVC,RMER2I     ;WAS "DVC" SET??
10481  050514  001424                              BEQ     8$              ;NO!!
10482  050516  013737  001370  001140              MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
10483  050524  042737  000200  001140              BIC     #DVC,$GDDAT
10484  050532  013737  001370  001142              MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10485  050540  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10486  050544  112776  000077  000000              MOVB    #77,@(SP)       ;WRITE ERROR NUMBER
10487  050552  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10488  050556  004736                              JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10489  050560  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
10490  050564  000240                              NOP
10491  050566                                  8$:
10492
10493                                          ;SEE IF "PIP" AND "OM" ARE 0, AND "ATA","MOL" AND "VV" ARE 1
10494  050566  013746  001340                      MOV     RMDSI,-(SP)     ;PUT RMDS ON STACK
10495  050572  042716  047676                      BIC     #^C<PIP!MOL!VV!OM!ATA>,(SP)
10496  050576  022726  110100                      CMP     #ATA!MOL!VV,(SP)+
10497  050602  001002                              BNE     85$
10498  050604  000137  051220                      JMP     13$
10499  050610                                  85$:
10500
10501                                          ;REPORT AN ERROR IF MOL = 0 AND OPI = 0, I.E., MEDIUM WENT OFF
10502                                          ;LINE AFTER RECALIBRATE WAS INITIATED
10503  050610  032737  010000  001340                BIT     #MOL,RMDSI      ;DID MOL DROP??
10504  050616  001030                              BNE     9$              ;NO!!
10505  050620  032737  020000  001342              BIT     #OPI,RMER1I     ;WAS OPI ERROR REPORTED??
10506  050626  001024                              BNE     9$              ;YES - DON'T REPORT MOL=0
10507  050630  013737  001340  001140              MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10508  050636  052737  010000  001140              BIS     #MOL,$GDDAT
10509  050644  013737  001340  001142              MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10510  050652  062716  000004                      ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10511  050656  112776  000100  000000              MOVB    #100,@(SP)      ;WRITE ERROR NUMBER
10512  050664  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10513  050670  004736                              JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10514  050672  162716  000010                      SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
10515  050676  000240                              NOP
10516  050700                                  9$:
10517
10518                                          ;REPORT AN ERROR IF "VV" = 0 AND "IVC" = 0
```

```
10519  050700  032737  000100  001340          BIT     #VV,RMDSI       ;DID "VV" NOP??
10520  050706  001030                          BNE     10$             ;NO!!
10521  050710  032737  010000  001370          BIT     #IVC,RMER2I     ;WAS THERE A IVC ERROR??
10522  050716  001024                          BNE     10$             ;YES - DONT REPORT VV=0
10523  050720  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10524  050726  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10525  050734  052737  000100  001140          BIS     #VV,$GDDAT
10526  050742  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10527  050746  112776  000101  000000          MOVB    #101,@(SP)      ;WRITE ERROR NUMBER IN CALL
10528  050754  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10529  050760  004736                          JSR     PC,@(SP)+
10530  050762  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
10531  050766  000240                          NOP
10532  050770                          10$:
10533
10534                                  ;REPORT AN ERROR IF ATA IS NOT SET
10535  050770  032737  100000  001340          BIT     #ATA,RMDSI      ;WAS ATA SET DURING RECALIBRATE??
10536  050776  001024                          BNE     11$             ;YES!!
10537  051000  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10538  051006  052737  100000  001140          BIS     #ATA,$GDDAT
10539  051014  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10540  051022  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10541  051026  112776  000102  000000          MOVB    #102,@(SP)      ;WRITE ERROR NUMBER IN CALL
10542  051034  162716  000002                  SUB     #2,(SP)
10543  051040  004736                          JSR     PC,@(SP)+
10544  051042  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
10545  051046  000240                          NOP
10546
10547  051050                          11$:
10548
10549                                  ;REPORT AN ERROR IF "OM" IS NOT ZERO BECAUSE RECALIBRATE SHOULD
10550                                  ;ALWAYS CLEAR OFFSET MODE
10551  051050  032737  000001  001340          BIT     #OM,RMDSI       ;WAS "OM" RESET??
10552  051056  001424                          BEQ     12$             ;YES!!
10553  051060  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10554  051066  042737  000001  001140          BIC     #OM,$GDDAT
10555  051074  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10556  051102  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10557  051106  112776  000103  000000          MOVB    #103,@(SP)      ;WRITE ERROR NUMBER
10558  051114  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10559  051120  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10560  051122  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
10561  051126  000240                          NOP
10562  051130                          12$:
10563
10564                                  ;REPORT AN ERROR IF "PIP" IS STIL ON, I.E., DRIVE NOT ON
10565                                  ;CYLINDER
10566  051130  032737  020000  001340          BIT     #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
10567  051136  001430                          BEQ     13$             ;NO!!
10568  051140  032737  040000  001370          BIT     #SKI,RMER2I     ;WAS "SKI" DETECTED??
10569  051146  001024                          BNE     13$             ;YES-DONT REPORT "PIP"
10570  051150  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
10571  051156  042737  020000  001140          BIC     #PIP,$GDDAT
10572  051164  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
10573  051172  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10574  051176  112776  000104  000000          MOVB    #104,@(SP)      ;WRITE ERROR NUMBER
```

```
10575  051204  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10576  051210  004736                  JSR     PC,@(SP)+
10577  051212  162716  000010          SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
10578  051216  000240                  NOP
10579  051220                  13$:
10580
10581                          ;SEE IF "ILR" OR "RMR" OR "UNS" IS SET
10582  051220  032737  040006  001342  BIT     #ILR!RMR!UNS,RMER1I
10583  051226  001514                  BEQ     16$
10584
10585                          ;REPORT AN ERROR IF "ILR" IS SET
10586  051230  032737  000002  001342  BIT     #ILR,RMER1I     ;WAS ILR SET DURING RECALIBRATE??
10587  051236  001424                  BEQ     14$             ;NO!!
10588  051240  013737  001342  001140  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10589  051246  042737  000002  001140  BIC     #ILR,$GDDAT
10590  051254  013737  001342  001142  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10591  051262  062716  000004          ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
10592  051266  112776  000105  000000  MOVB    #105,@(SP)      ;WRITE ERROR NUMBER IN CALL
10593  051274  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10594  051300  004736                  JSR     PC,@(SP)+
10595  051302  162716  000010          SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
10596  051306  000240                  NOP
10597  051310                  14$:
10598
10599                          ;REPORT AN ERROR IF "RMR" IS SET
10600  051310  032737  000004  001342  BIT     #RMR,RMER1I     ;WAS RMR SET??
10601  051316  001424                  BEQ     15$             ;NO!!
10602  051320  013737  001342  001140  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10603  051326  042737  000004  001140  BIC     #RMR,$GDDAT
10604  051334  013737  001342  001142  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10605  051342  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10606  051346  112776  000106  000000  MOVB    #106,@(SP)      ;WRITE ERROR NUMBER IN USER'S CALL
10607  051354  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10608  051360  004736                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10609  051362  162716  000010          SUB     #10,(SP)        ;MOVE SP TO USER'S BRANCH
10610  051366  000240                  NOP
10611  051370                  15$:
10612
10613                          ;REPORT AN ERROR IF "UNS" IS SET AND "DVC" IS 0
10614  051370  032737  040000  001342  BIT     #UNS,RMER1I     ;WAS UNSAFE ON"?
10615  051376  001430                  BEQ     16$             ;NO!!
10616  051400  032737  000200  001370  BIT     #DVC,RMER2I     ;WAS THERE A DEVICE CHECK??
10617  051406  001024                  BNE     16$             ;YES - DON'T REPORT UNSAFE
10618  051410  013737  001342  001140  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10619  051416  042737  040000  001140  BIC     #UNS,$GDDAT
10620  051424  013737  001342  001142  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10621  051432  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10622  051436  112776  000107  000000  MOVB    #107,@(SP)      ;WRITE ERROR NUMBER
10623  051444  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10624  051450  004736                  JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
10625  051452  162716  000010          SUB     #10,(SP)        ;MOVE SP BACK TO USER'S BRANCH
10626  051456  000240                  NOP
10627  051460                  16$:
10628
10629                          ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
10630  051460  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
```

```
10631  051464  105776  000000          TSTB    @(SP)           ;WAS AN ERROR REPORTED??
10632  051470  001403                  BEQ     17$             ;NO!!
10633  051472  062716  000004          ADD     #4,(SP)         ;YES - AUGMENT SP RETURN
10634  051476  000402                  BR      18$
10635  051500  162716  000004  17$:     SUB     #4,(SP) ;NO ERROR - RETURN SP TO BRANCH
10636  051504  000240          18$:     NOP
10637  051506  000207                  RTS     PC              ;STATUS CECK IS COMPLETE
10638
```

```
10639                                           .SBTTL  DRIVE CLEAR STATUS CHECK SUBROUTINE
10640                                           ;       BR      ???             RETURN HERE IF NO ERROR
10641                                           ;       NOP                     RETURN HERE TO REPORT AN ERROR
10642                                           ;       ERROR                   ERROR NUMBER DEFINED BY SUB
10643                                           ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
10644                                           ;       ???                     RETURN HERE IF NO MORE ERRORS
10645
10646  051510                                   DRVSTS:
10647
10648                                           ;CLEAR USER'S ERROR CALL
10649  051510  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10650  051514  105076  000000                          CLRB    @(SP)           ;CLEAR ERROR CALL
10651  051520  162716  000004                          SUB     #4,(SP)         ;MOVE SP TO USER'S BRANCH
10652                                           ;REPORT ERROR IF RMCS1 NOT INITIALIZED
10653  051524  013737  001326  001142    4$:   MOV     RMCS1I,$BDDAT   ;CHECK RMCS1
10654  051532  042737  173700  001142          BIC     #^C<DVA!FNCMSK>,$BDDAT  ;CLEAR DONT CARES
10655  051540  012737  004010  001140          MOV     #DVA!DRVCLR,$GDDAT      ;EXPECT DVA
10656  051546  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10657  051554  001443                          BEQ     6$              ;BRANCH IF EQUAL
10658  051556  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10659  051562  112776  000141  000000          MOVB    #141,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10660  051570  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10661  051574  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10662  051576  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10663  051602  000240                          NOP
10664                                           ;REPORT ERROR IF RMDS NOT INITIALIZED
10665  051604  013737  001340  001142    5$:   MOV     RMDSI,$BDDAT    ;CHECK RMDS
10666  051612  042737  021101  001142          BIC     #PGM!OM!VV!PIP,$BDDAT   ;CLEAR DONT CARES
10667  051620  012737  010600  001140          MOV     #MOL!DPR!DRY,$GDDAT     ;EXPECT DRY & DPR
10668  051626  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10669  051634  001413                          BEQ     6$              ;BRANCH IF EQUAL
10670  051636  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10671  051642  112776  000142  000000          MOVB    #142,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10672  051650  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10673  051654  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10674  051656  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10675  051662  000240                          NOP
10676                                           ;REPORT ERROR IF RMER1 NOT INITIALIZED
10677  051664  005037  001140            6$:   CLR     $GDDAT          ;EXPECT 0'S
10678  051670  013737  001342  001142          MOV     RMER1I,$BDDAT   ;CHECK RMER1
10679  051676  001413                          BEQ     8$              ;BRANCH IF EQUAL
10680  051700  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10681  051704  112776  000143  000000          MOVB    #143,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10682  051712  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10683  051716  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10684  051720  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10685  051724  000240                          NOP
10686                                           ;REPORT ERROR IF ATA NOT INITIALIZED
10687  051726  013737  001344  001142    8$:   MOV     RMASI,$BDDAT    ;CHECK ATTENTION BIT
10688  051734  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10689  051736  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
10690  051740  013701  001446                  MOV     TSTQUE,R1
10691  051744  116102  000001                  MOVB    1(R1),R2
10692  051750  042702  177400                  BIC     #^CATNMSK,R2
10693  051754  005102                          COM     R2
10694  051756  040237  001142                  BIC     R2,$BDDAT
```

```
10695  051762  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
10696  051764  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
10697  051766  005737  001142                  TST     $BDDAT          ;IS ATTENTION CLEARED??
10698  051772  001413                          BEQ     9$              ;BRANCH IF ATTENTION CLEARED
10699  051774  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10700  052000  112776  000144  000000          MOVB    #144,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10701  052006  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10702  052012  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10703  052014  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10704  052020  000240                          NOP
10705                                  ;REPORT ERROR IF RMMR1 NOT INITIALIZED
10706  052022  013737  001352  001142  9$:     MOV     RMMR1I,$BDDAT   ;CHECK RMMR
10707  052030  042737  000046  001142          BIC     #WC!LS!LST,$BDDAT       ;CLEAR DONT CARES
10708  052036  012737  000010  001140          MOV     #MWD,$GDDAT     ;EXPECT WRITE DATA ON
10709  052044  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
10710  052052  001413                          BEQ     11$             ;BRANCH IF ZERO
10711  052054  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10712  052060  112776  000145  000000          MOVB    #145,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10713  052066  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10714  052072  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10715  052074  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10716  052100  000240                          NOP
10717                                  ;REPORT ERROR IF RMMR2 NOT INITIALIZED
10718  052102  013737  001366  001142  11$:     MOV     RMMR2I,$BDDAT   ;CHECK RMMR2
10719  052110  042737  140000  001142          BIC     #RQA!RQB,$BDDAT ;CLEAR REQA, REQB
10720  052116  012737  011777  001140          MOV     #TST!1777,$GDDAT        ;EXPECT TEST BIT ON
10721  052124  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED & RECEIVED
10722  052132  001413                          BEQ     15$             ;BRANCH IF EQUAL
10723  052134  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10724  052140  112776  000146  000000          MOVB    #146,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10725  052146  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10726  052152  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10727  052154  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10728  052160  000240                          NOP
10729  052162  005037  001140          15$:    CLR     $GDDAT          ;EXPECT ZEROS
10730                                  ;REPORT ERROR IF RMEC2 NOT RESET
10731  052166  013737  001374  001142          MOV     RMEC2I,$BDDAT   ;CHECK RMEC2
10732  052174  001413                          BEQ     17$             ;BRANCH IF 0
10733  052176  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10734  052202  112776  000150  000000          MOVB    #150,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10735  052210  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10736  052214  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10737  052216  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10738  052222  000240                          NOP
10739                                  ;REPORT ERROR IF RMER2 NOT RESET
10740  052224  013737  001370  001142  17$:     MOV     RMER2I,$BDDAT   ;CHECK RMER2
10741  052232  001413                          BEQ     18$             ;BRANCH IF NO ERROR
10742  052234  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10743  052240  112776  000147  000000          MOVB    #147,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
10744  052246  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
10745  052252  004736                          JSR     PC,@(SP)+       ;REPORT THE ERROR VIA USER
10746  052254  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
10747  052260  000240                          NOP
10748  052262                          18$:
10749
10750  052262                          19$:
```

```
10751
10752                                   ;AUGMENT RETURN ADDRESS IF ANY ERROR WAS FOUND
10753  052262  062716  000004                ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
10754  052266  105776  000000                TSTB    @(SP)           ;WAS AN ERROR DETECTED??
10755  052272  001403                        BEQ     21$             ;NO!!
10756  052274  062716  000004                ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
10757  052300  000402                        BR      23$
10758  052302  162716  000004        21$:    SUB     #4,(SP) ;MOVE SP BACK TO NO ERROR RETURN
10759  052306  000240                23$:    NOP
10760  052310  000207                        RTS     PC              ;RETURN TO USER
```

```
                                        .SBTTL  DATA TRANSFER COMMAND STATUS CHECK SUBROUTINE

                                        ;THIS SUBROUTINE VERIFIES THE RESULTS OF ALL DATA TRANSFER COMMANDS
                                        ;USING STATUS STORED IN THE GET BUFFER AND TEST PARAMETERS
                                        ;STORED IN THE PUT BUFFER.  ERRORS ARE REPORTED BY WRITING
                                        ;THE ERROR NUMBER IN THE USERS ERROR CALL.

                                        ;USER'S SUBROUTINE CALL:
                                        ;(1)    JSR     PC,DTASTS
                                        ;(2)    BR      ??              RETURN HERE IF NO DATA ERRORS
                                        ;(3)    NOP                     RETURN HERE TO REPORT AN ERROR
                                        ;(4)    ERROR                   SUB WRITES ERROR NUMBER
                                        ;(5)    JSR     PC,@(SP)+       USER RETURNS FOR MORE CHECKS
                                        ;(6)    ??                      SUB RETURNS HERE AFTER ALL
                                        ;                               ERRORS ARE REPORTED

052312                                  DTASTS:

                                        ;CLEAR USER'S ERROR CALL AND ERROR FLAGS
052312  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
052316  105076  000000                          CLRB    @(SP)           ;CLEAR LOW ORDER BYTE OF TRAP
052322  162716  000004                          SUB     #4,(SP)         ;RESTORE SP TO NO ERROR
052326  005037  055706                          CLR     500$            ;CLEAR ERROR FLAGS

                                        ;REPORT ANY CONTROL BUS PARITY ERROR WHILE READING REMOTE REGISTERS, I.E., MCPE = 1
052332  032737  020000  001326                  BIT     #MCPE,RMCS1I    ;WAS THERE A PARITY ERROR??
052340  001422                                  BEQ     10$             ;NO!!
052342  013737  001326  001140                  MOV     RMCS1I,$GDDAT   ;EXPECTED STATUS
052350  042737  020000  001140                  BIC     #MCPE,$GDDAT
052356  013737  001326  001142                  MOV     RMCS1I,$BDDAT   ;RECEIVED STATUS
052364  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
052370  112776  000013  000000                  MOVB    #13,@(SP)       ;WRITE ERROR NUMBER
052376  162716  000002                          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
052402  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
052404  000466                                  BR      30$
052406                                  10$:

                                        ;REPORT ANY CONTROL BUS PARITY ERROR WHILE WRING REMOTE REGISTERS, I.E.,
                                        ;PAR=1  AND DPE = 0
052406  032737  000010  001342                  BIT     #PAR,RMER1I     ;WAS THERE A PARITY ERROR??
052414  001435                                  BEQ     20$             ;NO!!
052416  032737  000010  001370                  BIT     #DPE,RMER2I     ;WAS IT DUE TO CONTROL BUS??
052424  001031                                  BNE     20$             ;NO!!
052426  013737  001342  001140                  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
052434  042737  000010  001140                  BIC     #PAR,$GDDAT
052442  013737  001342  001142                  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
052450  062716  000004                          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
052454  112776  000050  000000                  MOVB    #50,@(SP)       ;WRITE ERROR NUMBER
052462  032737  001000  001356                  BIT     #MXF,RMCS1I     ;DID MXF GET SET??
052470  001003                                  BNE     15$             ;YES!!
052472  112776  000274  000000                  MOVB    #274,@(SP)      ;NO - CHANGE ERROR NUMBER
052500  162716  000002                  15$:    SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
052504  004736                                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
052506  000425                                  BR      30$

052510                                  20$:
```

```
10817
10818                                   ;LOOK FOR ANY ERRORS WHICH MAY HAVE OCCURRED DURING COMMAND INITIATION OR
10819                                   ;MECHANICAL POSITIONING
10820
10821                                   ;FIRST TEST MXF WHICH WOULD INDICATE COMPOSITE ERROR SET WHEN FUNCTION
10822                                   ;CODE AND GO BIT WERE LOADED
10823  052510  032737  001000  001336          BIT     #MXF,RMCS2I     ;WAS "MISSED TRANSFER" SET??
10824  052516  001425                        BEQ     40$             ;NO!!
10825  052520  013737  001336  001140        MOV     RMCS2I,$GDDAT   ;EXPECTED STATUS
10826  052526  042737  001000  001140        BIC     #MXF,$GDDAT
10827  052534  013737  001336  001142        MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
10828  052542  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10829  052546  112776  000275  000000        MOVB    #275,@(SP)      ;WRITE ERROR NUMBER
10830  052554  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10831  052560  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10832  052562                          30$:
10833
10834                                   ;RESTORE SP TO NO ERROR RETURN AND BYPASS FURHTER STATUS CHECKING
10835  052562  162716  000010                SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10836  052566  000137  055660                JMP     380$            ;SKIP TO END OF SUB
10837
10838  052572                          40$:
10839
10840                                   ;REPORT AN ERROR IF "OPI" ERROR OCCURRED DUE TO "MOL" = 0, OR IF "OPI"
10841                                   ;AND "MOL" ARE SET, BUT "VV" IS RESET, INDICATING AN INTERMITTENT
10842                                   ;"MOL"
10843  052572  032737  020000  001342          BIT     #OPI,RMER1I     ;IS "OPI" SET??
10844  052600  001447                        BEQ     60$             ;NO!!
10845  052602  013737  001342  001140        MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10846  052610  042737  020000  001140        BIC     #OPI,$GDDAT
10847  052616  013737  001342  001142        MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10848  052624  032737  010000  001340        BIT     #MOL,RMDSI      ;WAS MEDIUM OFF LINE??
10849  052632  001404                        BEQ     45$             ;YES'!
10850  052634  032737  000100  001340        BIT     #VV,RMDSI       ;WAS "MOL" INTERMITTENT??
10851  052642  001013                        BNE     50$             ;NO!!
10852  052644  062716  000004          45$:  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10853  052650  112776  000276  000000        MOVB    #276,@(SP)      ;WRITE ERROR NUMBER IN CALL
10854  052656  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10855  052662  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10856  052664  162716  000010                SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10857  052670  000413                        BR      60$
10858  052672                          50$:
10859
10860                                   ;REPORT "OPI" ERROR, WHICH IS DUE TO "ON CYLINDER" NOT DROPPING OR
10861                                   ;"RUN" TIMEOUT (20 MS) OR SEARCH TIMEOUT (50 MS)
10862  052672  062716  000004                ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10863  052676  112776  000277  000000        MOVB    #277,@(SP)      ;WRITE ERROR NUMBER IN CALL
10864  052704  162716  000002                SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10865  052710  004736                        JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10866  052712  162716  000010                SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10867  052716  000240                        NOP
10868  052720                          60$:
10869
10870                                   ;LOOK FOR "IVC" ERROR DURING COMMAND INITIATION
10871  052720  032737  010000  001370          BIT     #IVC,RMER2I     ;WAS THERE AN "IVC" ERROR??
10872  052726  001432                        BEQ     70$             ;NO!!
```

```
10873                                   ;REPORT "IVC" ERROR DUE TO "VV" = 0, OR REPORT ERRONEOUS "IVC" ERROR
10874  052730  013737  001370  001140           MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
10875  052736  042737  010000  001140           BIC     #IVC,$GDDAT
10876  052744  013737  001370  001142           MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10877  052752  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
10878  052756  112776  000300  000000           MOVB    #300,@(SP)      ;WRITE ERROR NUMBER IN CALL
10879  052764  032737  000100  001340           BIT     #VV,RMDSI       ;WAS VOLUME VALID??
10880  052772  001403                           BEQ     65$             ;NO!!
10881  052774  112776  000301  000000           MOVB    #301,@(SP)      ;CHANGE ERROR NUMBER
10882  053002  162716  000002           65$:    SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10883  053006  004736                           JSR     PC,@(SP)+       ;REPORT "IVC" ERROR AND RETURN
10884  053010  162716  000010                   SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10885  053014                           70$:
10886
10887                                   ;SEE IF "ILF" OR "RMR" IS SET
10888  053014  032737  000007  001342           BIT     #ILR!ILF!RMR, RMER1I
10889  053022  001510                           BEQ     100$            ;NO ERRORS DETECTED
10890                                   ;REPORT AN ERROR IF "ILR" IS SET
10891  053024  032737  000002  001342           BIT     #ILR,RMER1I     ;WAS "ILR" DETECTED??
10892  053032  001424                           BEQ     80$             ;NO!!
10893  053034  013737  001342  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10894  053042  042737  000002  001140           BIC     #ILR,$GDDAT
10895  053050  013737  001342  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10896  053056  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10897  053062  112776  000302  000000           MOVB    #302,@(SP)      ;WRITE ERROR NUMBER IN CALL
10898  053070  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10899  053074  004736                           JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10900  053076  162716  000010                   SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10901  053102  000240                           NOP
10902  053104                           80$:
10903
10904                                   ;REPORT AN ERROR IF "ILF" IS SET
10905  053104  032737  000001  001342           BIT     #ILF,RMER1I     ;WAS "ILF" DETECTED??
10906  053112  001424                           BEQ     90$             ;NO!!
10907  053114  013737  001342  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10908  053122  042737  000001  001140           BIC     #ILF,$GDDAT
10909  053130  013737  001342  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10910  053136  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10911  053142  112776  000303  000000           MOVB    #303,@(SP)      ;WRITE ERROR NUMBER IN CALL
10912  053150  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10913  053154  004736                           JSR     PC,@(SP)+       ;REPORT ERORR AND RETURN
10914  053156  162716  000010                   SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
10915  053162  000240                           NOP
10916  053164                           90$:
10917
10918                                   ;REPORT AN ERROR IF "RMR" IS SET
10919  053164  032737  000004  001342           BIT     #RMR,RMER1I     ;WAS "RMR" DETECTED??
10920  053172  001424                           BEQ     100$            ;NO!!
10921  053174  013737  001342  001140           MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
10922  053202  042737  000004  001140           BIC     #RMR,$GDDAT
10923  053210  013737  001342  001142           MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
10924  053216  062716  000004                   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10925  053222  112776  000304  000000           MOVB    #304,@(SP)      ;WRITE ERROR NUMBER IN CALL
10926  053230  162716  000002                   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10927  053234  004736                           JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10928  053236  162716  000010                   SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
```

```
10929  053242  000240                          NOP
10930  053244                          100$:
10931                                  ;DETERMINE WHETHER OR NOT "IAE" SHOULD BE SET AND CHECK FOR ERROR
10932  053244  012737  002000  001140          MOV     #IAE,$GDDAT     ;SETUP FOR "IAE" = 1
10933  053252  052737  040000  055706          BIS     #SKI,$OOS       ;SET SKI FLAG
10934  053260  022737  001466  001432          CMP     #822.,RMDCO     ;IS CYLINDER > 822??
10935  053266  103425                          BLO     110$            ;YES!!
10936  053270  042737  040000  055706          BIC     #SKI,$OOS       ;RESET SKI FLAG
10937  053276  122737  000004  001405          CMPB    #4,RMDAO+1      ;IS TRACK > 4??
10938  053304  103416                          BLO     110$            ;YES!!
10939  053306  122737  000037  001404          CMPB    #31.,RMDAO      ;IS SECTOR > 31??
10940  053314  103412                          BLO     110$            ;YES!!
10941  053316  122737  000035  001404          CMPB    #29.,RMDAO      ;IS SECTOR > 29??
10942  053324  103004                          BHIS    105$            ;NO - IAE SHOULD BE ZERO
10943  053326  032737  010000  001430          BIT     #FMT16,RMOFO    ;18 BIT FORMAT??
10944  053334  001402                          BEQ     110$            ;YES!!
10945  053336  005037  001140          105$:   CLR     $GDDAT          ;IAE SHOULD BE ZERO
10946  053342  013737  001342  001142  110$:   MOV     RMER1I,$BDDAT   ;GET RECEIVED STATUS
10947  053350  042737  175777  001142          BIC     #^CIAE,$BDDAT
10948  053356  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;IS "IAE" STATUS OK??
10949  053364  001004                          BNE     115$            ;NO!!
10950  053366  042737  040000  055706          BIC     #SKI,$OOS       ;IAE OK - SKI SHOULD BE 0
10951  053374  000412                          BR      120$
10952  053376  062716  000004          115$:   ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10953  053402  112776  000305  000000          MOVB    #305,@(SP)      ;WRITE ERROR NUMBER
10954  053410  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10955  053414  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10956  053416  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10957  053422                          120$:
10958
10959                                  ;REPORT AN ERROR IF "SKI" IS SET AND "IAE" STATUS WAS OK
10960  053422  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
10961  053430  042737  137777  001142          BIC     #^CSKI,$BDDAT
10962  053436  013737  055706  001140          MOV     $OOS,$GDDAT     ;EXPECTED STATUS
10963  053444  042737  137777  001140          BIC     #^CSKI,$GDDAT
10964  053452  032737  040000  001370          BIT     #SKI,RMER2I     ;WAS "SKI" SET??
10965  053460  001417                          BEQ     140$            ;NO!!
10966  053462  032737  040000  055706          BIT     #SKI,$OOS       ;WAS SKI CAUSED BY IAE = 0??
10967  053470  001032                          BNE     150$            ;YES - DON'T REPORT SKI
10968  053472  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
10969  053476  112776  000306  000000          MOVB    #306,@(SP)      ;WRITE ERROR NUMBER
10970  053504  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10971  053510  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10972  053512  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
10973  053516  000417                          BR      150$
10974
10975  053520                          140$:
10976
10977                                  ;REPORT AN ERROR IF SKI = 0 AND IAE WAS NOT DETECTED
10978  053520  032737  040000  055706          BIT     #SKI,$OOS       ;SHOULD SKI BE SET??
10979  053526  001413                          BEQ     150$            ;NO!!
10980  053530  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
10981  053534  112776  000307  000000          MOVB    #307,@(SP)      ;WRITE ERROR NUMBER IN CALL
10982  053542  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
10983  053546  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
10984  053550  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
```

```
10985  053554  000240                          NOP
10986  053556                          150$:
10987
10988                                  ;LOOK FOR "LSC" OR "LBC" OR "DVC" IN ERROR REGISTER #2
10989  053556  032737  006200  001370          BIT     #LSC!LBC!DVC,RMER2I
10990  053564  001512                          BEQ     180$            ;NO ERRORS SET
10991
10992                                  ;REPORT ANY DEVICE FAULT, I.E., "DVC" = 1
10993  053566  032737  000200  001370          BIT     #DVC,RMER2I     ;IS "DVC" = 1??
10994  053574  001424                          BEQ     160$            ;NO!!
10995  053576  013737  001370  001140          MOV     RMER2I,$GDOAT   ;EXPECTED STATUS
10996  053604  042737  000200  001140          BIC     #DVC,$GDOAT
10997  053612  013737  001370  001142          MOV     RMER2I,$BDOAT   ;RECEIVED STATUS
10998  053620  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
10999  053624  112776  000310  000000          MOVB    #310,2(SP)      ;WRITE ERROR NUMBER IN CALL
11000  053632  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11001  053636  004736                          JSR     PC,2(SP)+       ;REPORT ERROR AND RETURN
11002  053640  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
11003  053644  000240                          NOP
11004  053646                          160$:
11005
11006                                  ;REPORT LOSS OF BIT CLOCK, I.E.; "LBC" = 1, IF "MOL" = 1
11007  053646  032737  002000  001370          BIT     #LBC,RMER2I     ;IS LBC SET??
11008  053654  001430                          BEQ     170$            ;NO!!
11009  053656  032737  010000  001340          BIT     #MOL,RMDSI      ;WAS LBC ERROR BY MOL = 0
11010  053664  001424                          BEQ     170$            ;YES!!
11011  053666  013737  001370  001140          MOV     RMER2I,$GDOAT   ;EXPECTED STATUS
11012  053674  042737  002000  001140          BIC     #LBC,$GDOAT
11013  053702  013737  001370  001142          MOV     RMER2I,$BDOAT   ;RECEIVED STATUS
11014  053710  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11015  053714  112776  000311  000000          MOVB    #311,2(SP)      ;WRITE ERROR NUMBER IN CALL
11016  053722  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11017  053726  004736                          JSR     PC,2(SP)+       ;REPORT ERROR AND RETURN
11018  053730  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
11019  053734  000240                          NOP
11020  053736                          170$:
11021
11022                                  ;REPORT LOS OF SYSTEM CLOCK, I.E., "LSC" = 1
11023  053736  032737  004000  001370          BIT     #LSC,RMER2I     ;IS "LSC" = 1??
11024  053744  001422                          BEQ     180$            ;NO!!
11025  053746  013737  001370  001140          MOV     RMER2I,$GDOAT   ;EXPECTED STATUS
11026  053754  042737  004000  001140          BIC     #LSC,$GDOAT
11027  053762  013737  001370  001142          MOV     RMER2I,$BDOAT   ;RECEIVED STATUS
11028  053770  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11029  053774  112776  000312  000000          MOVB    #312,2(SP)      ;WRITE ERROR NUMBER
11030  054002  004736                          JSR     PC,2(SP)+       ;REPORT ERROR AND RETURN
11031  054004  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
11032  054010  000240                          NOP
11033  054012                          180$:
11034
11035                                  ;LOOK FOR "UNS" OR "DTE" OR "WLE" IN ERROR REGISTER #1
11036  054012  032737  054000  001342          BIT     #UNS!DTE!WLE,RMER1I
11037  054020  001527                          BEQ     220$            ;NO BITS SET
11038                                  ;REPORT "UNS" ERROR IF "DVC" = 0
11039  054022  032737  040000  001342          BIT     #UNS,RMER1I     ;I_ "UNS" SET??
11040  054030  001427                          BEQ     190$            ;NO!!
```

```
11041  054032  032737  000200  001370          BIT     #DVC,RMER2I     ;WAS "UNS" CAUSED BY "DVC"??
11042  054040  001023                          BNE     190$            ;YES!!
11043  054042  013737  001342  001140          MOV     RMER1I,$GDOAT   ;EXPECTED STATUS
11044  054050  042737  040000  001140          BIC     #UNS,$GDOAT
11045  054056  013737  001342  001142          MOV     RMER1I,$BDOAT   ;RECEIVED STATUS
11046  054064  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11047  054070  112776  000313  000000          MOVB    #313,2(SP)      ;WRITE ERROR NUMBER
11048  054076  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11049  054102  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11050  054104  162716  000010                  SUB     #10,(SP)        ;RESTORE SP TO NO ERROR
11051  054110                          190$:
11052
11053                                  ;REPORT ANY DRIVE TIMING ERROR, I.E., "DTE" = 1
11054  054110  032737  010000  001342          BIT     #DTE,RMER1I     ;IS DTE SET??
11055  054116  001423                          BEQ     200$            ;NO!!
11056  054120  013737  001342  001140          MOV     RMER1I,$GDOAT   ;EXPECTED STATUS
11057  054126  042737  010000  001140          BIC     #DTE,$GDOAT
11058  054134  013737  001342  001142          MOV     RMER1I,$BDOAT   ;RECEIVED STATUS
11059  054142  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11060  054146  112776  000314  000000          MOVB    #314,2(SP)      ;WRITE ERROR NUMBER IN CALL
11061  054154  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11062  054160  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11063  054162  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11064  054166                          200$:
11065
11066                                  ;REPORT AN ERROR IF WRITE LOCK ERROR IS SET.  SEE IF DRIVE IS NOT
11067                                  ;WRITE PROTECTED, OR IF FUNCTION WAS NOT A WRITE
11068  054166  032737  004000  001342          BIT     #WLE,RMER1I     ;WAS "WLE" SET??
11069  054174  001441                          BEQ     220$            ;NO!!
11070  054176  013737  001342  001142          MOV     RMER1I,$BDOAT   ;RECEIVED STATUS
11071  054204  013737  001342  001140          MOV     RMER1I,$GDOAT   ;EXPECTED STATUS
11072  054212  052737  004000  001140          BIS     #WLE,$GDOAT
11073  054220  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11074  054224  112776  000315  000000          MOVB    #315,2(SP)      ;WRITE ERROR NUMBER IN CALL
11075  054232  032737  004000  001340          BIT     #WRL,RMDSI      ;WAS DRIVE WRITE PROTECTED??
11076  054240  001404                          BEQ     205$            ;NO!!
11077  054242  032737  000010  001376          BIT     #BIT3,RMCS10    ;WAS COMMAND A WRITE??
11078  054250  001406                          BEQ     210$            ;YES!!
11079  054252  112776  000316  000000  205$:   MOVB    #316,2(SP)      ;CHANGE ERROR NUMBER
11080  054260  042737  004000  001140          BIC     #WLE,$GDOAT
11081  054266  162716  000002          210$:   SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11082  054272  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11083  054274  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11084
11085  054300                          220$:
11086
11087                                  ;OMIT DATA ERROR CHECKS IF ANY PREVIOUS ERRORS HAVE BEEN DETECTED
11088  054300  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
11089  054304  105776  000000                  TSTB    @(SP)           ;WAS ERROR DETECTED??
11090  054310  001404                          BEQ     225$            ;NO - DO DATA CHECKS
11091  054312  162716  000004                  SUB     #4,(SP)         ;RESTORE SP
11092  054316  000137  055320                  JMP     340$            ;SKIP DATA CHECKS
11093  054322  162716  000004          225$:   SUB     #4,(SP)         ;RESTORE SP
11094
11095                                  ;CHECK HEADER ERRORS IF FUNCTION WAS NOT WRITE HEADER AND DATA, AND
11096                                  ;IF HEADER COMPARE IS NOT INHIBITED
```

```
11097  054326  013737  001376  055710          MOV     RMCS10,S10$     ;STRIP AND STORE FUNCTION CODE
11098  054334  042737  177700  055710          BIC     #^CFNCMSK,S10$
11099  054342  022737  000063  055710          CMP     #WH!GO,S10$     ;WAS FUNCTION WRITE HEADER & DATA??
11100  054350  001512                          BEQ     250$            ;YES - SKIP HEADER CHECKS
11101  054352  032737  002000  001360          BIT     #HCI,RMOFI      ;WAS HCI SET??
11102  054360  001106                          BNE     250$            ;YES - SKIP HEADER CHECKS
11103
11104                                  ;SEE IF ANY HEADER ERRORS ARE SET, I.E., "FER" OR "HCRC" OR "HCE"
11105  054362  032737  000620  001342          BIT     #HCRC!FER!HCE,RMER1I
11106  054370  001533                          BEQ     270$            ;NO ERRORS SET
11107
11108                                  ;REPORT HEADER CRC ERROR IF SET
11109  054372  032737  000400  001342          BIT     #HCRC,RMER1I    ;WAS HCRC SET??
11110  054400  001422                          BEQ     230$            ;NO!!
11111  054402  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11112  054410  042737  000400  001140          BIC     #HCRC,$GDDAT
11113  054416  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11114  054424  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
11115  054430  112776  000317  000000          MOVB    #317,@(SP)      ;WRITE ERROR NUMBER
11116  054436  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11117  054442  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11118  054444  000501                          BR      260$
11119  054446                          230$:
11120
11121                                  ;REPORT FORMAT ERROR IF SET
11122  054446  032737  000020  001342          BIT     #FER,RMER1I     ;WAS "FER" SET??
11123  054454  001422                          BEQ     240$            ;NO!!
11124  054456  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11125  054464  042737  000020  001140          BIC     #FER,$GDDAT
11126  054472  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11127  054500  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USERS ERROR
11128  054504  112776  000320  000000          MOVB    #320,@(SP)      ;WRITE ERROR NUMBER
11129  054512  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11130  054516  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11131  054520  000453                          BR      260$
11132  054522                          240$:
11133
11134                                  ;REPORT HEADER COMPARE ERROR IF SET
11135  054522  032737  000200  001342          BIT     #HCE,RMER1I     ;WAS "HCE" SET??
11136  054530  001453                          BEQ     270$            ;NO!!
11137  054532  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11138  054540  042737  000200  001140          BIC     #HCE,$GDDAT
11139  054546  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11140  054554  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
11141  054560  112776  000321  000000          MOVB    #321,@(SP)      ;WRITE ERROR NUMBER
11142  054566  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11143  054572  004736                          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11144  054574  000425                          BR      260$
11145                                  ;THERE SHOULD BE NO HEADER ERRORS BECAUSE
11146                                  ;       .COMMAND WAS WRITE HEADER AND DATA, OR
11147                                  ;       .HEADER COMPARE INHIBIT WAS SET
11148  054576  032737  000620  001342  250$:   BIT     #HCE!FER!HCRC,RMER1I
11149  054604  001425                          BEQ     270$            ;NO ERRORS WERE SET
11150  054606  013737  001342  001140          MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11151  054614  042737  000620  001140          BIC     #HCE!FER!HCRC,$GDDAT
11152  054622  013737  001342  001142          MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
```

```
11153  054630  062716  000004                  ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR CALL
11154  054634  112776  000322  000000          MOVB    #322,@(SP)       ;WRITE ERROR NUMBER
11155  054642  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN IF ERROR
11156  054646  004736                          JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
11157  054650  162716  000010          260$:   SUB     #10,(SP)         ;MOVE SP TO NO ERROR
11158  054654  000137  055320                  JMP     340$             ;OMIT FURTHER DATA CHECKS
11159
11160  054660                          270$:
11161
11162                                  ;IF COMMAND WAS A WRITE COMMAND, GO DO WRITE ERROR CHECKS, OTHERWISE
11163                                  ;DO READ ERROR CHECKS
11164  054660  032737  000010  055710          BIT     #BIT3,$10$       ;WAS THIS A WRITE COMMAND?
11165  054666  001002                          BNE     275$             ;NO!!
11166  054670  000137  055106                  JMP     310$             ;GO DO WRITE STATUS CHECK
11167  054674                          275$:
11168
11169                                  ;REPORT DATA CHECK IF SET
11170  054674  032737  100000  001342          BIT     #DCK,RMER1I      ;DATA CHECK ERROR??
11171  054702  001450                          BEQ     290$             ;NO!!
11172  054704  013737  001342  001140          MOV     RMER1I,$GDDAT    ;EXPECTED STATUS
11173  054712  042737  100000  001140          BIC     #DCK,$GDDAT
11174  054720  013737  001342  001142          MOV     RMER1I,$BDDAT    ;RECEIVED STATUS
11175  054726  062716  000004                  ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR
11176  054732  112776  000323  000000          MOVB    #323,@(SP)       ;WRITE ERROR NUMBER
11177  054740  032737  004000  001360          BIT     #ECI,RMOFI       ;WAS ECC CORRECTION DISABLED??
11178  054746  001021                          BNE     280$             ;YES!!
11179  054750  112776  000324  000000          MOVB    #324,@(SP)       ;CHANGE TO RECOVERABLE ERROR
11180  054756  032737  000100  001342          BIT     #ECH,RMER1I      ;IS ERROR RECOVERABLE??
11181  054764  001007                          BNE     276$             ;NO !!
11182                                  ;DO NOT REPORT RECOVERABLE ERROR IF READ COMMAND
11183  054766  032737  000020  055710          BIT     #BIT4,$10$       ;WAS THIS A READ COMMAND ??
11184  054774  001406                          BEQ     280$             ;NO !!
11185  054776  162716  000004                  SUB     #4,(SP)          ;RESTORE SP
11186  055002  000410                          BR      290$             ;SKIP ERROR - DATA WILL BE CORRECTED
11187  055004  112776  000325  000000  276$:   MOVB    #325,@(SP)       ;CHANGE TO NON RECOVERABLE
11188  055012  162716  000002          280$:   SUB     #2,(SP)          ;MOVE SP TO RETURN IF ERROR
11189  055016  004736                          JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
11190  055020  162716  000010                  SUB     #10,(SP)         ;RESTORE SP TO NO ERROR
11191
11192  055024                          290$:
11193
11194                                  ;REPORT DATA BUS PARITY ERROR IF SET, I.E., MDPE = 1
11195  055024  032737  000400  001336          BIT     #MDPE,RMCS2I     ;PARITY ERROR SET??
11196  055032  001423                          BEQ     300$             ;NO!!
11197  055034  013737  001336  001140          MOV     RMCS2I,$GDDAT    ;EXPECTED STATUS
11198  055042  042737  000400  001140          BIC     #MDPE,$GDDAT
11199  055050  013737  001336  001142          MOV     RMCS2I,$BDDAT    ;RECEIVED STATUS
11200  055056  062716  000004                  ADD     #4,(SP)          ;MOVE SP TO USER'S ERROR
11201  055062  112776  000326  000000          MOVB    #326,@(SP)       ;WRITE ERROR NUMBER
11202  055070  162716  000002                  SUB     #2,(SP)          ;MOVE SP TO RETURN IF ERROR
11203  055074  004736                          JSR     PC,@(SP)+        ;REPORT ERROR AND RETURN
11204  055076  162716  000010                  SUB     #10,(SP)         ;MOVE SP TO NO ERROR
11205  055102  000137  055320          300$:   JMP     340$             ;SKIP WRITE STATUS CHECK
11206
11207  055106                          310$:
11208
```

```
11209
11210  055106  032737  000001  001340          ;TEST TO SEE THAT OFFSET MODE WAS RESET; REPORT ERROR IF "OM" = 1
11211  055114  001423                  BIT     #OM,RMDSI       ;IS OFFSET ON??
11212  055116  013737  001340  001140  BEQ     320$            ;NO
11213  055124  042737  000001  001140  MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11214  055132  013737  001340  001142  BIC     #OM,$GDDAT
11215  055140  062716  000004          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11216  055144  112776  000327  000000  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11217  055152  162716  000002          MOVB    #327,@(SP)      ;WRITE ERROR NUMBER IN CALL
11218  055156  004736                  SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11219  055160  162716  000010          JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11220  055164                          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11221                                  320$:
11222                                  ;TEST FOR DATA BUS PARITY ERROR; REPORT ERROR IF "DPE" = 1
11223  055164  032737  000010  001370          BIT     #DPE,RMER2I     ;DATA PARITY ERROR??
11224  055172  001423                  BEQ     330$            ;NO!!
11225  055174  013737  001370  001140  MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11226  055202  042737  000010  001140  BIC     #DPE,$GDDAT
11227  055210  013737  001370  001142  MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11228  055216  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11229  055222  112776  000330  000000  MOVB    #330,@(SP)      ;WRITE ERROR NUMBER
11230  055230  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11231  055234  004736                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11232  055236  162716  000010          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11233  055242                          330$:
11234
11235                                  ;TEST FOR WRITE CLOCK FAILURE; REPORT ERROR IF "WCF" = 1
11236  055242  032737  000040  001342          BIT     #WCF,RMER1I     ;IS "WCF" SET??
11237  055250  001423                  BEQ     340$            ;NO!!
11238  055252  013737  001342  001140  MOV     RMER1I,$GDDAT   ;EXPECTED STATUS
11239  055260  042737  000040  001140  BIC     #WCF,$GDDAT
11240  055266  013737  001342  001142  MOV     RMER1I,$BDDAT   ;RECEIVED STATUS
11241  055274  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11242  055300  112776  000331  000000  MOVB    #331,@(SP)      ;WRITE ERROR NUMBER
11243  055306  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11244  055312  004736                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11245  055314  162716  000010          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11246  055320                          340$:
11247
11248                                  ;REPORT "DATA LATE" ERROR IF "DLT" = 1
11249  055320  032737  100000  001336          BIT     #DLT,RMCS2I     ;IS "DLT" SET??
11250  055326  001423                  BEQ     350$            ;NO!!
11251  055330  013737  001336  001140  MOV     RMCS2I,$GDDAT   ;EXPECTED STATUS
11252  055336  042737  100000  001140  BIC     #DLT,$GDDAT
11253  055344  013737  001336  001142  MOV     RMCS2I,$BDDAT   ;RECEIVED STATUS
11254  055352  062716  000004          ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
11255  055356  112776  000332  000000  MOVB    #332,@(SP)      ;WRITE ERROR NUMBER
11256  055364  162716  000002          SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
11257  055370  004736                  JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
11258  055372  162716  000010          SUB     #10,(SP)        ;MOVE SP TO NO ERROR
11259  055376                          350$:
11260                                  ;LOOK FOR UNEXPECTED CHANGES IN DRIVE STATUS
11261  055376  013746  001340          MOV     RMDSI,-(SP)     ;STACK DRIVE STATUS
11262  055402  042716  147677          BIC     #^C<PIP!MOL!VV>,(SP) ;CLEAR DONT CARES
11263  055406  022726  010100          CMP     #MOL!VV,(SP)+   ;IS DRIVE STATUS OK??
11264  055412  001522                  BEQ     380$            ;YES!!
```

```
 11265
 11266                                          ;REPORT ERROR IF POSITIONING IN PROGRESS AND NO SEEK INCOMPLETE ERROR,
 11267                                          ;I.E, PIP = 1 AND SKI = 0
 11268  055414  032737  020000  001340                BIT     #PIP,RMDSI      ;IS "PIP" SET??
 11269  055422  001430                              BEQ     360$            ;NO!!
 11270  055424  032737  040000  001370              BIT     #SKI,RMER2I     ;WAS "SKI" ERROR REPORTED??
 11271  055432  001024                              BNE     360$            ;YES-DONT REPORT PIP
 11272  055434  013737  001340  001140              MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
 11273  055442  042737  020000  001140              BIC     #PIP,$GDDAT
 11274  055450  013737  001340  001142              MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
 11275  055456  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 11276  055462  112776  000333  000000              MOVB    #333,@(SP)      ;WRITE ERROR NUMBER
 11277  055470  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
 11278  055474  004736                              JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
 11279  055476  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO NO ERROR
 11280  055502  000240                              NOP
 11281  055504                                  360$:
 11282
 11283                                          ;REPORT ERROR IF MEDIUM IS NOT ON LINE AND OPI ERROR WAS NOT
 11284                                          ;REPORTED, I.E., MOL = OPI = 0
 11285  055504  032737  010000  001340                BIT     #MOL,RMDSI      ;IS MEDIUM ON LINE??
 11286  055512  001027                              BNE     370$            ;YES!!
 11287  055514  032737  020000  001342              BIT     #OPI,RMER1I     ;WAS OPI ERROR REPORTED??
 11288  055522  001023                              BNE     370$            ;YES!!
 11289  055524  013737  001340  001140              MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
 11290  055532  052737  010000  001140              BIS     #MOL,$GDDAT
 11291  055540  013737  001340  001142              MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
 11292  055546  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR
 11293  055552  112776  000334  000000              MOVB    #334,@(SP)      ;WRITE ERROR NUMBER
 11294  055560  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
 11295  055564  004736                              JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
 11296  055566  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO NO ERROR
 11297  055572                                  370$:
 11298
 11299                                          ;REPORT ERROR IF VOLUME IS NOT VALID AND "IVC" ERROR WAS NOT
 11300                                          ;REPORTED, I.E., VV = IVC = 0
 11301  055572  032737  000100  001340                BIT     #VV,RMDSI       ;IS VOLUME VALID??
 11302  055600  001027                              BNE     380$            ;YES!!
 11303  055602  032737  010000  001370              BIT     #IVC,RMER2I     ;WAS IVC ERROR REPORTED??
 11304  055610  001033                              BNE     390$            ;YES!!
 11305  055612  013737  001340  001140              MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
 11306  055620  052737  000100  001140              BIS     #VV,$GDDAT
 11307  055626  013737  001340  001142              MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
 11308  055634  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO USERS ERROR CALL
 11309  055640  112776  000335  000000              MOVB    #335,@(SP)      ;WRITE ERROR NUMBER
 11310  055646  162716  000002                      SUB     #2,(SP)         ;MOVE SP TO RETURN IF ERROR
 11311  055652  004736                              JSR     PC,@(SP)+       ;REPORT ERROR AND RETURN
 11312  055654  162716  000010                      SUB     #10,(SP)        ;MOVE SP TO NO ERROR
 11313  055660                                  380$:
 11314
 11315                                          ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS FOUND
 11316  055660  062716  000004                      ADD     #4,(SP)         ;MOVE SP TO ERROR CALL
 11317  055664  105776  000000                      TSTB    @(SP)           ;ANY ERROR??
 11318  055670  001403                              BEQ     390$            ;NO!!
 11319  055672  062716  000004                      ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
 11320  055676  000402                              BR      400$
```

```
 11321  055700  162716  000004          390$:   SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
 11322
 11323  055704  000207                  400$:   RTS     PC              ;RETURN TO USER
 11324
 11325  055706  000000                  500$:   .WORD                   ;ERROR FLAGS
 11326  055710  000000                  510$:   .WORD                   ;TEMPORARY STORAGE
```

```
                                .SBTTL  STATIC DRIVE STATUS CHECK SUBROUTINE

                                ;THIS SUBROUTINE LOOKS FOR UNEXPECTED CHANGES IN DRIVE
                                ;STATUS, SUCH AS THE DRIVE LOSING VOLUME VALID.  THE SUBROUTINE
                                ;CAN BE USED BY HOUSEKEEPING AND OTHER COMMANDS DURING WHICH THERE
                                ;SHOULD NOT BE ANY DRIVE ERRORS OR CHANGES IN STATE.

                                ;THE FOLLOWING CONDITIONS ARE TESTED AND REPORTED AS ERRORS
                                ;IF TRUE:

                                ;       .MOL = 0, INDICATES DRIVE WENT OFFLINE, NOTE
                                ;THAT MOL IS ASSUMED TO HAVE BEEN SET
                                ;       .VV = 0, INDICATES THE DRIVE LOST VOLUME VALID
                                ;       .PIP = 1, INDICATES THAT THE DRIVE IS OFF CYLINDER
                                ;       .SKI = 1, INDICATES THE DRIVE HAS AN UNEXPECTED SKI ERROR
                                ;       .DVC = 1, INDICATES AN UNEXPECTED DEVICE FAULT

                                ;THE SUBROUTINE IS CALLED AFTER STORING STATUS IN THE GET BUFFER.

                                ;(1)    JSR     PC,STCDRVSTS
                                ;       BR      ???             RETURN HERE IF NO ERROR
                                ;       NOP                     RETURN HERE TO REPORT AN ERROR
                                ;       ERROR                   ERROR NUMBER DEFINED BY SUB
                                ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
                                ;       ???                     RETURN HERE IF NO MORE ERRORS

055712                          STCDRVSTS:

                                ;CLEAR USER'S ERROR CALL
055712  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
055716  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
055722  162716  000004                  SUB     #4,(SP)         ;MOVE SP BACK TO NO ERROR RETURN
                                ;SEE IF "MOL" = "VV" = 1, AND "PIP" = 0
055726  013746  001340                  MOV     RMDSI,-(SP)     ;PUT DRIVE STATUS ON STACK
055732  042716  147677                  BIC     #^C<PIP!MOL!VV>,(SP)
055736  022726  010100                  CMP     #MOL!VV,(SP)+   ;ARE MOL,VV AND PIP O.K.??
055742  001524                          BEQ     30$             ;YES!!

                                ;REPORT AN ERROR IF MOL = 0 AND "OPI" = 0
055744  032737  010000  001340          BIT     #MOL,RMDSI      ;IS MOL ON ??
055752  001030                          BNE     10$             ;YES!!
055754  032737  020000  001342          BIT     #OPI,RMER1I     ;WAS "OPI" SET??
055762  001024                          BNE     10$             ;YES-DONT REPORT "MOL" = 0
055764  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
055772  052737  010000  001140          BIS     #MOL,$GDDAT
056000  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
056006  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
056012  112776  000207  000000          MOVB    #207,@(SP)      ;WRITE ERROR NUMBER IN CALL
056020  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
056024  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
056026  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR RETURN
```

```
11378  056032  000240                          NOP
11379  056034                          10$:
11380
11381                                  ;REPORT AN ERROR IF VOLUME VALID IS NOW ZERO AND "IVC" = 0
11382  056034  032737  000100  001340          BIT     #VV,RMDSI       ;IS "VV" = 0??
11383  056042  001030                          BNE     20$             ;NO!!
11384  056044  032737  010000  001370          BIT     #IVC,RMER2I     ;WAS "IVC" SET??
11385  056052  001024                          BNE     20$             ;YES-DONT REPORT "VV" = 0
11386  056054  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11387  056062  052737  000100  001340          BIS     #VV,RMDSI
11388  056070  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11389  056076  062716  000004                  ADD     #4,(SP) ;MOVE SP TO USER'S ERROR CALL
11390  056102  112776  000210  000000          MOVB    #210,@(SP)      ;WRITE ERROR NUMBER IN CALL
11391  056110  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11392  056114  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11393  056116  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11394  056122  000240                          NOP
11395  056124                          20$:
11396
11397                                  ;REPORT AN ERROR IF DRIVE IS OFF CYLINDER AND "SKI" = 0
11398  056124  032737  020000  001340          BIT     #PIP,RMDSI      ;IS DRIVE OFF CYLINDER??
11399  056132  001430                          BEQ     30$             ;NO!!
11400  056134  032737  040000  001370          BIT     #SKI,RMER2I     ;WAS "SKI" SET??
11401  056142  001024                          BNE     30$             ;YES-DONT REPORT "PIP" = 1
11402  056144  013737  001340  001140          MOV     RMDSI,$GDDAT    ;EXPECTED STATUS
11403  056152  042737  020000  001140          BIC     #PIP,$GDDAT
11404  056160  013737  001340  001142          MOV     RMDSI,$BDDAT    ;RECEIVED STATUS
11405  056166  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11406  056172  112776  000211  000000          MOVB    #211,@(SP)      ;WRITE ERROR NUMBER IN USER'S CALL
11407  056200  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11408  056204  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11409  056206  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11410  056212  000240                          NOP
11411  056214                          30$:
11412
11413                                  ;SEE IF "SKI" = "DVC" = 0
11414  056214  013746  001370                  MOV     RMER2I,-(SP)    ;PUT ERROR REG 2 ON STACK
11415  056220  042726  137577                  BIC     #^C<SKI!DVC>,(SP)+
11416  056224  001460                          BEQ     60$             ;BRANCH IF NO ERROR
11417  056226                          40$:
11418
11419                                  ;REPORT AN ERROR IF THERE IS A DEVICE FAULT
11420  056226  032737  000200  001370          BIT     #DVC,RMER2I     ;ANY DEVICE FAULT??
11421  056234  001424                          BEQ     50$             ;NO!!
11422  056236  013737  001370  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11423  056244  042737  000200  001140          BIC     #DVC,$GDDAT
11424  056252  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11425  056260  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S CALL
11426  056264  112776  000212  000000          MOVB    #212,@(SP)      ;WRITE NUMBER OF ERROR IN CALL
11427  056272  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11428  056276  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11429  056300  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11430  056304  000240                          NOP
11431  056306                          50$:
11432
11433                                  ;REPORT AN ERROR IF "SKI" = 1
```

```
11434  056306  032737  040000  001370          BIT     #SKI,RMER2I     ;IS THERE A SEEK INCOMPLETE ERROR
11435  056314  001424                          BEQ     60$             ;NO!!
11436  056316  013737  001370  001140          MOV     RMER2I,$GDDAT   ;EXPECTED STATUS
11437  056324  042737  040000  001140          BIC     #SKI,$GDDAT
11438  056332  013737  001370  001142          MOV     RMER2I,$BDDAT   ;RECEIVED STATUS
11439  056340  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11440  056344  112776  000213  000000          MOVB    #213,@(SP)      ;WRITE ERROR NUMBER IN USER'S ERROR CALL
11441  056352  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11442  056356  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11443  056360  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO NO ERROR
11444  056364  000240                          NOP
11445  056366                          60$:
11446
11447                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
11448  056366  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11449  056372  105776  000000                  TSTB    @(SP)           ;WAS AN ERROR DETECTED??
11450  056376  001403                          BEQ     70$             ;NO!!
11451  056400  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO USER'S ERROR RETURN
11452  056404  000402                          BR      80$
11453  056406  162716  000004          70$:    SUB     #4,(SP)         ;NO - MOVE SP TO NO ERROR RETURN
11454  056412  000240                  80$:    NOP
11455  056414  000207                          RTS     PC              ;RETURN TO USER
```

```
                                .SBTTL  COMPOSITE ERROR CHECK SUBROUTINE

                                ;THIS SUBROUTINE CHECKS THE STORED CONTENTS OF RMER1 AND
                                ;RMER2 AFTER MASKING EACH REGISTER WORD WITH THE USER'S STATUS
                                ;MASKS AND REPORTS AN ERROR IF ANY BITS ARE LEFT ON AFTER
                                ;THE MASKS ARE APPLIED.

                                ;CALL:
                                ;(1)    JSR     PC,CMPERRSTS
                                ;       .WORD                   MASK FOR ERROR REGISTER 1
                                ;       .WORD                   MASK FOR ERROR REGISTER 2
                                ;       BR      ???             RETURN HERE IF NO ERROR
                                ;       NOP                     RETURN HERE TO REPORT AN ERROR
                                ;       ERROR                   ERROR NUMBER DEFINED BY SUB
                                ;       JSR     PC,@(SP)+       GO BACK TO SUB FOR MORE ERROR CHECKS
                                ;       ???                     RETURN HERE IF NO MORE ERRORS

                                ;NOTE:  BITS TO BE MASKED SHOULD BE ONE; BITS TO BE TESTED SHOULD
                                ;BE ZERO

056416                          CMPERRSTS:

                                ;MASK AND STORE THE CONTENTS OF RMER1  AND RMER2
056416  013737  001342  001176          MOV     RMER1I,$TMP1    ;STORE RMER1 AT TEMP STORAGE
056424  047637  000000  001176          BIC     @(SP),$TMP1     ;MASK   RMER1
056432  062716  000002                  ADD     #2,(SP)         ;MOVE SP TO NEXT MASK
056436  013737  001370  001200          MOV     RMER2I,$TMP2    ;STORE RMER2 AT TEMP STORAGE
056444  047637  000000  001200          BIC     @(SP),$TMP2     ;MASK RMER2


                                ;CLEAR USER'S ERROR CALL
056452  062716  000006                  ADD     #6,(SP)         ;MOVE SP TO USER'S ERROR CALL
056456  105076  000000                  CLRB    @(SP)           ;CLEAR ERROR NUMBER
056462  162716  000004                  SUB     #4,(SP)         ;LEAVE SP AT NO ERROR RETURN

                                ;SEE IF THERE WERE ANY ERRORS IN RMER1, I.E., $TMP1
056466  005737  001176                  TST     $TMP1           ;ANY ERRORS TO REPORT??
056472  001420                          BEQ     5$              ;NO !!
056474  013737  001176  001142          MOV     $TMP1,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
056502  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
056506  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
056512  112776  000066  000000          MOVB    #66,@(SP)       ;CORRECTABLE DATA CHECK ERROR #
056520  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
056524  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
056526  162716  000010                  SUB     #10,(SP)        ;MOVE SP BACK TO BRANCH
056532  000240                          NOP
056534                          5$:

                                ;SEE IF THERE ARE ANY ERRORS TO REPORT IN RMER2 ($TMP2)
056534  005737  001200                  TST     $TMP2           ;ANY ERRORS IN RMER2?
056540  001420                          BEQ     10$             ;NO!!

056542  013737  001200  001142          MOV     $TMP2,$BDDAT    ;RECEIVED STATUS FOR TYPEOUT
056550  005037  001140                  CLR     $GDDAT          ;EXPECTED STATUS FOR TYPEOUT
056554  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
```

```
11512  056560  112776  000067  000000          MOVB    #67,@(SP)       ;WRITE ERROR NUMBER IN USER'S CALL
11513  056566  162716  000002                  SUB     #2,(SP)         ;MOVE SP TO RETURN FOR ERROR
11514  056572  004736                          JSR     PC,@(SP)+       ;REPORT ERROR VIA USER
11515  056574  162716  000010                  SUB     #10,(SP)        ;MOVE SP TO NO ERROR RETURN
11516  056600  000240                          NOP
11517  056602                          10$:
11518
11519                                  ;AUGMENT THE RETURN ADDRESS IF ANY ERROR WAS DETECTED
11520  056602  062716  000004                  ADD     #4,(SP)         ;MOVE SP TO USER'S ERROR CALL
11521  056606  105776  000000                  TSTB    @(SP)           ;WAS THERE AN ERROR CALLED??
11522  056612  001403                          BEQ     20$             ;NO!!
11523  056614  062716  000004                  ADD     #4,(SP)         ;YES - MOVE SP TO ERROR RETURN
11524  056620  000402                          BR      30$
11525  056622  162716  000004          20$:    SUB     #4,(SP)         ;MOVE SP TO NO ERROR RETURN
11526  056626  000207                  30$:    RTS     PC              ;RETURN TO USER
11527
11528
```

```
11529                                   .SBTTL  STOP AND SHUTDOWN SUBROUTINES
11530
11531  056630                           STOP:
11532
11533                                   ;DROP PRIORITY TO ALLOW CONSOLE INTERRUPT
11534  056630  012746  000140                   MOV     #PR3,-(SP)      ;;PUT NEW PS ON STACK
11535  056634  012746  056642                   MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
11536  056640  000002                           RTI                     ;;POP NEW PC AND PS
11537  056642                           64$:
11538  056642  000240                           NOP
11539                                   ;RAISE PRIORITY TO INHIBIT INTERRUPT
11540  056644  012746  000300                   MOV     #PR6,-(SP)      ;;PUT NEW PS ON STACK
11541  056650  012746  056656                   MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
11542  056654  000002                           RTI                     ;;POP NEW PC AND PS
11543  056656                           65$:
11544
11545  056656  000207                   10$:    RTS     PC              ;CONTINUE
11546
11547
11548  056660                           SHUT:
11549  056660  104401  056702                   TYPE    ,100$           ;TYPE THE HALT MESSAGE
11550  056664  005737  000042                   TST     42              ;RUNNING STANDALONE ??
11551  056670  001402                           BEQ     10$             ;YES !!
11552  056672  000137  032262                   JMP     $EOP            ;NO - GO TO END OF PASS
11553  056676                           10$:
11554  056676  000137  005322                   JMP     START           ;GO TO START
11555  056702  042524  052123  053440 100$:    .ASCIZ  @TEST WAS HALTED@<CR><LF><LF>
11556  056710  051501  044040  046101
11557  056716  042524  006504  005012
11558  056724     000
11559           056726                   .EVEN
```

```
11560                                   .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
11561
11562                                   ;;**********************************************************************
11563                                   ;*SAVE R0-R5
11564                                   ;*CALL:
11565                                   ;*      SAVREG
11566                                   ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
11567                                   ;*
11568                                   ;*TOP---(+16)
11569                                   ;* +2---(+18)
11570                                   ;* +4---R5
11571                                   ;* +6---R4
11572                                   ;* +8---R3
11573                                   ;*+10---R2
11574                                   ;*+12---R1
11575                                   ;*+14---R0
11576
11577  056726                           $SAVREG:
11578  056726  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
11579  056730  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11580  056732  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
11581  056734  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
11582  056736  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
11583  056740  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
11584  056742  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
11585  056746  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
11586  056752  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
11587  056756  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
11588  056762  000002                          RTI
11589
11590                                   ;*RESTORE R0-R5
11591                                   ;*CALL:
11592                                   ;*      RESREG
11593  056764                           $RESREG:
11594  056764  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
11595  056770  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
11596  056774  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
11597  057000  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
11598  057004  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
11599  057006  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
11600  057010  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
11601  057012  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
11602  057014  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11603  057016  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
11604  057020  000002                          RTI
11605                                   .SBTTL  BINARY TO ASCII AND TYPE ROUTINE
11606
11607                                   ;;**********************************************************************
11608                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
11609                                   ;*BINARY-ASCII NUMBER AND TYPE IT.
11610                                   ;*CALL:
11611                                   ;*      MOV     NUMBER,-(SP)    ;;NUMBER TO BE TYPED
11612                                   ;*      TYPBN                   ;;TYPE IT
11613
11614  057022  010146                   $TYPBN: MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
11615  057024  016601  000006                  MOV     6(SP),R1        ;;GET THE INPUT NUMBER
```

```
11616  057030  000261                          SEC                     ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
11617  057032  112737  000060  057074  1$:     MOVB    #'0,$BIN        ;;SET CHARACTER TO AN ASCII "0".
11618  057040  006101                          ROL     R1              ;;GET THIS BIT
11619  057042  001406                          BEQ     2$              ;;DONE?
11620  057044  105537  057074                  ADCB    $BIN            ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
11621  057050  104401  057074                  TYPE    ,$BIN           ;;GO TYPE THIS BIT
11622  057054  000241                          CLC                     ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
11623  057056  000765                          BR      1$              ;;GO DO THE NEXT BIT
11624  057060  012601                  2$:     MOV     (SP)+,R1        ;;POP THE STACK INTO R1
11625  057062  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
11626  057070  012616                          MOV     (SP)+,(SP)
11627  057072  000002                          RTI                     ;;RETURN TO USER
11628  057074      000      000         $BIN:   .BYTE   0,0             ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
11629                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
11630
11631                                  ;;**************************************************************
11632                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
11633                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
11634                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
11635                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
11636                                  ;*REPLACED WITH SPACES.
11637                                  ;*CALL:
11638                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
11639                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
11640
11641  057076                          $TYPDS:
11642  057076  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
11643  057100  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11644  057102  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
11645  057104  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
11646  057106  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
11647  057110  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
11648  057114  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
11649  057120  100004                          BPL     1$              ;;BR IF INPUT IS POS.
11650  057122  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
11651  057124  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
11652  057132  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
11653  057134  012703  057312                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
11654  057140  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
11655  057144  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
11656  057146  016001  057302                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
11657  057152  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
11658  057154  002402                          BLT     4$              ;;BR IF DONE
11659  057156  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
11660  057160  000774                          BR      3$
11661  057162  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
11662  057164  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
11663  057166  001002                          BNE     5$              ;;FALL THROUGH IF 0
1.664  057170  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
11665  057172  100407                          BMI     7$              ;;BR IF YES
11666  057174  106316                  5$:     ASLB    (SP)            ;;MSD?
11667  057176  103003                          BCC     6$              ;;BR IF NO
11668  057200  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
11669  057206  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
11670  057212  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
1.671  057216  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
```

```
11672  057220  005720                          TST     (R0)+           ;;JUST INCREMENTING
11673  057222  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
11674  057226  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
11675  057230  003002                          BGT     8$              ;;GO TO EXIT
11676  057232  010502                          MOV     R5,R2           ;;GET THE LSD
11677  057234  000746                          BR      6$              ;;GO CHANGE TO ASCII
11700  057236  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
11701  057240  100003                          BPL     9$              ;;BR IF NO
11702  057242  116662  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
11703  057250  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
11704  057252  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
11705  057254  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
11706  057256  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
11707  057260  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11710  057262  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
11711  057264  104401  057312                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
11712  057270  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
11713  057276  012616                          MOV     (SP)+,(SP)
11714  057300  000002                          RTI                     ;;RETURN TO USER
11715  057302  023420                  $DTBL:  10000.
11716  057304  001750                          1000.
11717  057306  000144                          100.
11720  057310  000012                          10.
11721  057312  000004                  $DBLK:  .BLKW   4
11722                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
11723
11724                                  ;;**********************************************************************
11725                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
11726                                  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
11727                                  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
11730                                  ;*CALL:
11731                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
11732                                  ;*      TYPOS                   ;;CALL FOR TYPEOUT
11733                                  ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
11734                                  ;*      .BYTE   M               ;;M=1 OR 0
11735                                  ;*                                      ;;1=TYPE LEADING ZEROS
11736                                  ;*                                      ;;0=SUPPRESS LEADING ZEROS
11737                                  ;*
11740                                  ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
11741                                  ;*$TYPOS OR $TYPOC
11742                                  ;*CALL:
11743                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
11744                                  ;*      TYPON                   ;;CALL FOR TYPEOUT
11745                                  ;*
11746                                  ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
11747                                  ;*CALL:
11750                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
11751                                  ;*      TYPOC                   ;;CALL FOR TYPEOUT
11752
11753  057322  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
11754  057326  116637  000001  057545          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
11755  057334  112637  057547                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
11756  057340  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
11757  057344  000406                          BR      $TYPON
11760  057346  112737  000001  057545  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
11761  057354  112737  000006  057547          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
```

```
11728  057362  112737  000005  057544  STYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
11729  057370  010346                          MOV     R3,-(SP)        ;;SAVE R3
11730  057372  010446                          MOV     R4,-(SP)        ;;SAVE R4
11731  057374  010546                          MOV     R5,-(SP)        ;;SAVE R5
11732  057376  113704  057547                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
11733  057402  005404                          NEG     R4
11734  057404  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
11735  057410  110437  057546                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
11736  057414  113704  057545                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
11737  057420  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
11738  057424  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
11739  057426  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
11740  057430  000404                          BR      3$              ;;GO DO MSB
11741  057432  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
11742  057434  006105                          ROL     R5
11743  057436  006105                          ROL     R5
11744  057440  010503                          MOV     R5,R3
11745  057442  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
11746  057444  105337  057546                  DECB    $OMODE          ;;TYPE THIS DIGIT?
11747  057450  100016                          BPL     7$              ;;BR IF NO
11748  057452  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
11749  057456  001002                          BNE     4$              ;;TEST FOR 0
11750  057460  005704                          TST     R4              ;;SUPPRESS THIS 0?
11751  057462  001403                          BEQ     5$              ;;BR IF YES
11752  057464  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
11753  057466  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
11754  057472  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
11755  057476  110337  057542                  MOVB    R3,8$           ;;SAVE FOR TYPING
11756  057502  104401  057542                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
11757  057506  105337  057544          7$:     DECB    $OCNT           ;;COUNT BY 1
11758  057512  003347                          BGT     2$              ;;BR IF MORE TO DO
11759  057514  002402                          BLT     6$              ;;BR IF DONE
11760  057516  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
11761  057520  000744                          BR      2$              ;;GO DO THE LAST DIGIT
11762  057522  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
11763  057524  012604                          MOV     (SP)+,R4        ;;RESTORE R4
11764  057526  012603                          MOV     (SP)+,R3        ;;RESTORE R3
11765  057530  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
11766  057536  012616                          MOV     (SP)+,(SP)
11767  057540  000002                          RTI                     ;;RETURN
11768  057542     000          8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
11769  057543     000                  .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
11770  057544     000          $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
11771  057545     000          $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
11772  057546  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
11773                          .SBTTL  TYPE ROUTINE
11774
11775                          ;;**************************************************************
11776                          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
11777                          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
11778                          ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
11779                          ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
11780                          ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
11781                          ;*
11782                          ;*CALL:
11783                          ;*1) USING A TRAP INSTRUCTION
```

```
11784                                   ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
11785                                   ;*OR
11786                                   ;*      TYPE
11787                                   ;*      MESADR
11788                                   ;*
11789
11790  057550  105737  001173          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
11791  057554  100002                          BPL     1$              ;;BR IF YES
11792  057556  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
11793  057560  000430                          BR      3$              ;;LEAVE
11794  057562  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
11795  057564  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
11796  057570  122737  000001  001242          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
11797  057576  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
11798  057600  132737  000100  001243          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
11799  057606  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
11800  057610  010037  057620                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
11801  057614  004737  062474                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
11802  057620  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
11803  057622  132737  000040  001243  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
11804  057630  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
11805  057632  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
11806  057634  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
11807  057636  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
11808  057640  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
11809  057642  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
11810  057646  000002                          RTI                     ;;RETURN
11811  057650  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
11812  057654  001430                          BEQ     8$
11813  057656  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
11814  057662  001006                          BNE     5$
11815  057664  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
11816  057666  104401                          TYPE                    ;;TYPE A CR AND LF
11817  057670  001217                          $CRLF
11818  057672  105037  060026                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
11819  057676  000755                          BR      2$              ;;GET NEXT CHARACTER
11820  057700  004737  057762          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
11821  057704  123726  001172          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
11822  057710  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
11823  057712  013746  001170                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
11824                                                                  ;;AND THE NULL CHAR.
11825  057716  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
11826  057722  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
11827  057724  004737  057762                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
11828  057730  105337  060026                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
11829  057734  000770                          BR      7$              ;;LOOP
11830
11831                                   ;HORIZONTAL TAB PROCESSOR
11832
11833  057736  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
11834  057742  004737  057762          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
11835  057746  132737  000007  060026          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
11836  057754  001372                          BNE     9$              ;;TAB STOP
11837  057756  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
11838  057760  000724                          BR      2$              ;;GET NEXT CHARACTER
11839  057762  105777  121176          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
```

```
11840  057766  100375                        BPL     $TYPEC
11841  057770  116677  000002  121170        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
11842  057776  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
11843  060004  001003                        BNE     1$              ;;BRANCH IF NO
11844  060006  105037  060026                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
11845  060012  000406                        BR      $TYPEX          ;;EXIT
11846  060014  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
11847  060022  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
11848  060024  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
11849  060026  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
11850  060030  000207          $TYPEX: RTS     PC
11851
11852                           .SBTTL  SCOPE HANDLER ROUTINE
11853
11854                           ;;**************************************************************
11855                           ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
11856                           ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
11857                           ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
11858                           ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11859                           ;*SW14=1        LOOP ON TEST
11860                           ;*SW11=1        INHIBIT ITERATIONS
11861                           ;*SW09=1        LOOP ON ERROR
11862                           ;*SW08=1        LOOP ON TEST IN SWR<7:0>
11863                           ;*CALL
11864                           ;*      SCOPE           ;;SCOPE=IOT
11865
11866  060032                   $SCOPE:
11867  060032  104410                        CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
11868  060034  004737  056630                JSR     PC,STOP
11869  060040  032777  040000  121106  1$:   BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
11870  060046  001131                        BNE     $OVER           ;;YES IF SW14=1
11871                           ;#####START OF CODE FOR THE XOR TESTER#####
11872  060050  000416          $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
11873                                                          ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
11874  060052  013746  000004                MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
11875  060056  012737  060076  000004        MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
11876  060064  005737  177060                TST     @#177060        ;;TIME OUT ON XOR?
11877  060070  012637  000004                MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
11878  060074  000500                        BR      $SVLAD          ;;GO TO THE NEXT TEST
11879  060076  022626          5$:   CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
11880  060100  012637  000004                MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
11881  060104  000440                        BR      7$              ;;LOOP ON THE PRESENT TEST
11882  060106                  6$:;#####END OF CODE FOR THE XOR TESTER#####
11883  060106  032777  000400  121040        BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
11884  060114  001421                        BEQ     2$              ;;BR IF NO
11885  060116  005046                        CLR     -(SP)           ;;CLEAR A TEMP. LOCATION
11886  060120  117716  121030                MOVB    @SWR,(SP)       ;;PICKUP THE DESIRED TEST NUMBER
11887  060124  001414                        BEQ     8$              ;;BRANCH IF BAD TEST NUMBER IN SWR
11888  060126  022716  000035                CMP     #35,(SP)        ;;CHECK THE NUMBER IN THE SWR
11889  060132  002411                        BLT     8$              ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
11890  060134  011637  001116                MOV     (SP),$TSTNM     ;;UPDATE THE TEST NUMBER
11891  060140  005316                        DEC     (SP)            ;;BACKUP BY ONE
11892  060142  006316                        ASL     (SP)            ;;SCALE THE TEST NUMBER AS AN INDEX
11893  060144  062716  060350                ADD     #$SWR08TBL,(SP) ;;FORM THE ADDRESS OF TEST POINTER
11894  060150  013637  001122                MOV     @(SP)+,$LPADR   ;;SET LOOP ADDRESS TO DESIRED TEST
11895  060154  000466                        BR      $OVER           ;;GO LOOP ON THE TEST
```

```
 11896  060156  005726                  8$:     TST     (SP)+           ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
 11897  060160  105737  001117          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
 11898  060164  001421                          BEQ     3$              ;;BR IF NO
 11899  060166  123737  001131  001117          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
 11900  060174  101015                          BHI     3$              ;;BR IF NO
 11901  060176  032777  001000  120750          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
 11902  060204  001404                          BEQ     4$              ;;BR IF NO
 11903  060206  013737  001124  001122  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
 11904  060214  000446                          BR      $OVER
 11905  060216  105037  001117          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
 11906  060222  005037  001206                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
 11907  060226  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
 11908  060230  032777  004000  120716  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
 11909  060236  001011                          BNE     1$              ;;BR IF YES
 11910  060240  005737  001230                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
 11911  060244  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
 11912  060246  005237  001120                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
 11913  060252  023737  001206  001120          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
 11914  060260  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
 11915  060262  012737  000001  001120  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
 11916  060270  013737  060346  001206          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
 11917  060276  105237  001116          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
 11918  060302  113737  001116  001226          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
 11919  060310  011637  001122                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
 11920  060314  011637  001124                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
 11921  060320  005037  001210                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 11922  060324  112737  000001  001131          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
 11923  060332  013777  001116  120616  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
 11924  060340  013716  001122                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 11925  060344  000002                          RTI                     ;;FIXES PS
 11926  060346  000012                  $MXCNT: 10.                     ;;MAX. NUMBER OF ITERATIONS
 11927  060350                          $SWOBTBL:
 11928  060350  007020                          .WORD   TST1+2          ;;STARTING ADDRESS OF TEST 1
 11929  060352  007216                          .WORD   TST2+2          ;;STARTING ADDRESS OF TEST 2
 11930  060354  007402                          .WORD   TST3+2          ;;STARTING ADDRESS OF TEST 3
 11931  060356  007544                          .WORD   TST4+2          ;;STARTING ADDRESS OF TEST 4
 11932  060360  010344                          .WORD   TST5+2          ;;STARTING ADDRESS OF TEST 5
 11933  060362  011144                          .WORD   TST6+2          ;;STARTING ADDRESS OF TEST 6
 11934  060364  011760                          .WORD   TST7+2          ;;STARTING ADDRESS OF TEST 7
 11935  060366  012534                          .WORD   TST10+2         ;;STARTING ADDRESS OF TEST 10
 11936  060370  013454                          .WORD   TST11+2         ;;STARTING ADDRESS OF TEST 11
 11937  060372  014254                          .WORD   TST12+2         ;;STARTING ADDRESS OF TEST 12
 11938  060374  015030                          .WORD   TST13+2         ;;STARTING ADDRESS OF TEST 13
 11939  060376  015746                          .WORD   TST14+2         ;;STARTING ADDRESS OF TEST 14
 11940  060400  016522                          .WORD   TST15+2         ;;STARTING ADDRESS OF TEST 15
 11941  060402  017276                          .WORD   TST16+2         ;;STARTING ADDRESS OF TEST 16
 11942  060404  020052                          .WORD   TST17+2         ;;STARTING ADDRESS OF TEST 17
 11943  060406  020654                          .WORD   TST20+2         ;;STARTING ADDRESS OF TEST 20
 11944  060410  021400                          .WORD   TST21+2         ;;STARTING ADDRESS OF TEST 21
 11945  060412  022124                          .WORD   TST22+2         ;;STARTING ADDRESS OF TEST 22
 11946  060414  022646                          .WORD   TST23+2         ;;STARTING ADDRESS OF TEST 23
 11947  060416  023144                          .WORD   TST24+2         ;;STARTING ADDRESS OF TEST 24
 11948  060420  023442                          .WORD   TST25+2         ;;STARTING ADDRESS OF TEST 25
 11949  060422  024170                          .WORD   TST26+2         ;;STARTING ADDRESS OF TEST 26
 11950  060424  024716                          .WORD   TST27+2         ;;STARTING ADDRESS OF TEST 27
 11951  060426  025444                          .WORD   TST30+2         ;;STARTING ADDRESS OF TEST 30
```

```
 11952  060430  026336                          .WORD   TST31+2         ;;STARTING ADDRESS OF TEST 31
 11953  060432  027144                          .WORD   TST32+2         ;;STARTING ADDRESS OF TEST 32
 11954  060434  027666                          .WORD   TST33+2         ;;STARTING ADDRESS OF TEST 33
 11955  060436  030410                          .WORD   TST34+2         ;;STARTING ADDRESS OF TEST 34
 11956  060440  031416                          .WORD   TST35+2         ;;STARTING ADDRESS OF TEST 35
 11957                                  .SBTTL  ERROR HANDLER ROUTINE
 11958
 11959                                  ;;**************************************************************
 11960                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
 11961                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
 11962                                  ;*AND GO TO ERRTYP ON ERROR
 11963                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 11964                                  ;*SW15=1         HALT ON ERROR
 11965                                  ;*SW13=1         INHIBIT ERROR TYPEOUTS
 11966                                  ;*SW10=1         BELL ON ERROR
 11967                                  ;*SW09=1         LOOP ON ERROR
 11968                                  ;*CALL
 11969                                  ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
 11970
 11971  060442                          $ERROR:
 11972  060442  104410                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
 11973  060444  105237  001117          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
 11974  060450  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
 11975  060452  013777  001116  120476          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
 11976  060460  032777  002000  120466          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
 11977  060466  001402                          BEQ     1$              ;;NO - SKIP
 11978  060470  104401  001212                  TYPE    ,$BELL          ;;RING BELL
 11979  060474  005237  001126          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
 11980  060500  011637  001132                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
 11981  060504  162737  000002  001132          SUB     #2,$ERRPC
 11982  060512  117737  120414  001130          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
 11983  060520  032777  020000  120426          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
 11984  060526  001004                          BNE     20$             ;;SKIP TYPEOUTS
 11985  060530  004737  032472                  JSR     PC,ERRTYP       ;;GO TO USER ERROR ROUTINE
 11986  060534  104401  001217                  TYPE    ,$CRLF
 11987  060540                          20$:
 11988  060540  122737  000001  001242          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
 11989  060546  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
 11990  060550  113737  001130  060562          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
 11991  060556  004737  062504                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
 11992  060562     000                  21$:    .BYTE   0
 11993  060563     000                          .BYTE   0
 11994  060564  000777                  22$:    BR      22$             ;;APT ERROR LOOP
 11995  060566  005777  120362          2$:     TST     @SWR            ;;HALT ON ERROR
 11996  060572  100002                          BPL     3$              ;;SKIP IF CONTINUE
 11997  060574  000000                          HALT                    ;;HALT ON ERROR!
 11998  060576  104410                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
 11999  060600  032777  001000  120346  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
 12000  060606  001402                          BEQ     4$              ;;BR IF NO
 12001  060610  013716  001124                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
 12002  060614  005737  001210          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
 12003  060620  001402                          BEQ     5$              ;;BR IF NONE
 12004  060622  013716  001210                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
 12005  060626                          5$:
 12006  060626  022737  032452  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
 12007  060634  001001                          BNE     6$              ;;BRANCH IF NO
```

```
12008  060636  000000                          HALT                    ;;YES
12009  060640
12010  060640  000002                  6$:     RTI                     ;;RETURN
12011                                  .SBTTL  TTY INPUT ROUTINE
12012
12013                                  ;;*****************************************************************
12014                                  .ENABL  LSB
12015  060642  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
12016  060644  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
12017  060646  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
12018  060650  000001                  $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
12019          060651                  $TKQEND=.
12020          060652                  .EVEN
12021
12022                                  ;*TK INITIALIZE ROUTINE
12023                                  ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
12024                                  ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
12025                                  ;
12026                                  ;*CALL:
12027                                  ;*      JSR     PC,$TKINT
12028                                  ;*      RETURN
12029                                  ;
12030  060652  005037  060642          $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
12031  060656  012737  060650  060644          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
12032  060664  013737  060644  060646          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
12033  060672  012737  060722  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
12034  060700  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
12035  060706  005777  120250                  TST     @$TKB           ;;CLEAR DONE FLAG
12036  060712  012777  000100  120240          MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
12037  060720  000207                          RTS     PC              ;;RETURN TO CALLER
12038
12039                                  ;*TK SERVICE ROUTINE
12040                                  ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
12041                                  ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
12042                                  ;*IT IN THE QUEUE.
12043                                  ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
12044                                  ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (SHUT)
12045                                  ;
12046  060722  117746  120234          $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
12047  060726  042716  177600                  BIC     #^C177,(SP)     ;;STRIP THE JUNK
12048  060732  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL C?
12049  060736  001007                          BNE     1$              ;;BRANCH IF NO
12050  060740  104401  062036                  TYPE    $CNTLC          ;;TYPE A CONTROL-C (^C)
12051  060744  004737  060652                  JSR     PC,$TKINT       ;;INIT THE KEYBOARD
12052  060750  005726                          TST     (SP)+           ;;CLEAN UP STACK
12053  060752  000137  056660                  JMP     SHUT            ;;CONTROL C RESTART
12054  060756  021627  000007          1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
12055  060762  001004                          BNE     2$              ;;BRANCH IF NO
12056  060764  022737  000176  001154          CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
12057  060772  001500                          BEQ     6$              ;;GO TO SWR CHANGE
12058
12059  060774                          2$:
12060  060774  022737  000001  060642          CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
12061  061002  001004                          BNE     3$              ;;BRANCH IF NO
12062  061004  104401  001212                  TYPE    $BELL           ;;RING THE TTY BELL
12063  061010  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
```

```
12064  061012  000451                        BR      5$              ;;EXIT
12065  061014  021627  000023          3$:   CMP     (SP),#23        ;;IS IT A CONTROL-S?
12066  061020  001021                        BNE     32$             ;;BRANCH IF NO
12067  061022  005077  120132                CLR     @STKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
12068  061026  005726                        TST     (SP)+           ;;CLEAN CHAR OFF STACK
12069  061030  105777  120124          31$:  TSTB    @STKS           ;;WAIT FOR A CHAR
12070  061034  100375                        BPL     31$             ;;LOOP UNTIL ITS THERE
12071  061036  117746  120120                MOVB    @STKB,-(SP)     ;;GET THE CHARACTER
12072  061042  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
12073  061046  022627  000021                CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
12074  061052  001366                        BNE     31$             ;;BRANCH IF NO
12075  061054  012777  000100  120076        MOV     #100,@STKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
12076  061062  000002                        RTI                     ;;RETURN
12077  06..  4  005237  060642          32$:  INC     $TKCNT          ;;COUNT THIS CHARACTER
12078  061070  021627  000140                CMP     (SP),#140       ;;IS IT UPPER CASE?
12079  061074  002405                        BLT     4$              ;;BRANCH IF YES
12080  061076  021627  000175                CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
12081  061102  003002                        BGT     4$              ;;BRANCH IF YES
12082  061104  042716  000040                BIC     #40,(SP)        ;;MAKE IT UPPER CASE
12083  061110  112677  177530          4$:   MOVB    (SP)+,@STKQIN   ;;AND PUT IT IN QUEUE
12084  061114  005237  060644                INC     $TKQIN          ;;UPDATE THE POINTER
12085  061120  023727  060644  060651        CMP     $TKQIN,#STKQEND ;;GO OFF THE END?
12086  061126  001003                        BNE     5$              ;;BRANCH IF NO
12087  061130  012737  060650  060644        MOV     #STKQSRT,$TKQIN ;;RESET THE POINTER
12088  061136  000002                  5$:   RTI                     ;;RETURN
12089
12090                                  ;;**********************************************************************
12091                                  ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
12092                                  ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
12093                                  ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
12094                                  ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
12095  061140  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
12096  061146  001124                        BNE     15$             ;;EXIT IF NOT
12097  061150  105777  120004                TSTB    @STKS           ;;IS A CHAR WAITING?
12098  061154  100121                        BPL     15$             ;;IF NOT, EXIT
12099  061156  117746  120000                MOVB    @STKB,-(SP)     ;;YES
12100  061162  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
12101  061166  021627  000007                CMP     (SP),#7         ;;IS IT A CONTROL-G?
12102  061172  001300                        BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
12103                                                                 ;;AND EXIT
12104
12105                                  ;;**********************************************************************
12106                                  ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
12107                                  ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
12108                                  ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
12109  061174  123727  001150  000001  6$:   CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
12110  061202  001674                        BEQ     2$              ;;BRANCH IF YES
12111  061204  005726                        TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
12112  061206  004737  060652                JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
12113  061212  005077  117742                CLR     @STKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
12114  061216  112737  000001  001151        MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
12115
12116  061224  104401  062050                TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
12117  061230  104401  062055          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
12118  061234  013746  000176                MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
12119  061240  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
```

```
12120  061242  104401  062066            TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
12121  061246  005046                19$: CLR     -(SP)           ;;CLEAR COUNTER
12122  061250  005046                     CLR     -(SP)           ;;THE NEW SWR
12123  061252  105777  117702        7$: TSTB    @$TKS           ;;CHAR THERE?
12124  061256  100375                     BPL     7$              ;;IF NOT TRY AGAIN
12125
12126  061260  117746  117676            MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
12127  061264  042716  177600            BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
12128
12129  061270  021627  000003            CMP     (SP),#3         ;;IS IT A CONTROL-C?
12130  061274  001015                     BNE     9$              ;;BRANCH IF NOT
12131  061276  104401  062036            TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
12132  061302  062706  000006            ADD     #6,SP           ;;CLEAN UP STACK
12133  061306  123727  001151  000001    CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
12134  061314  001003                     BNE     8$              ;;BRANCH IF NO
12135  061316  012777  000100  117634    MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
12136  061324  000137  056660        8$: JMP     SHUT            ;;CONTROL-C RESTART
12137
12138
12139  061330  021627  000025        9$: CMP     (SP),#25        ;;IS IT A CONTROL-U?
12140  061334  001005                     BNE     10$             ;;BRANCH IF NOT
12141  061336  104401  062043            TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
12142  061342  062706  000006       20$: ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
12143  061346  000737                     BR      19$             ;;LET'S TRY IT AGAIN
12144
12145
12146  061350  021627  000015       10$: CMP     (SP),#15        ;;IS IT A <CR>?
12147  061354  001022                     BNE     16$             ;;BRANCH IF NO
12148  061356  005766  000004            TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
12149  061362  001403                     BEQ     11$             ;;BRANCH IF YES
12150  061364  016677  000002  117562    MOV     2(SP),@SWR      ;;SAVE NEW SWR
12151  061372  062706  000006       11$: ADD     #6,SP           ;;CLEAR UP STACK
12152  061376  104401  001217       14$: TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
12153  061402  123727  001151  000001    CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
12154  061410  001003                     BNE     15$             ;;BRANCH IF NOT
12155  061412  012777  000100  117540    MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
12156  061420  000002       15$: RTI                             ;;RETURN
12157  061422  004737  057762       16$: JSR     PC,$TYPEC       ;;ECHO CHAR
12158  061426  021627  000060            CMP     (SP),#60        ;;CHAR < 0?
12159  061432  002420                     BLT     18$             ;;BRANCH IF YES
12160  061434  021627  000067            CMP     (SP),#67        ;;CHAR > 7?
12161  061440  003015                     BGT     18$             ;;BRANCH IF YES
12162  061442  042726  000060            BIC     #60,(SP)+       ;;STRIP-OFF ASCII
12163  061446  005766  000002            TST     2(SP)           ;;IS THIS THE FIRST CHAR
12164  061452  001403                     BEQ     17$             ;;BRANCH IF YES
12165  061454  006316                     ASL     (SP)            ;;NO, SHIFT PRESENT
12166  061456  006316                     ASL     (SP)            ;;   CHAR OVER TO MAKE
12167  061460  006316                     ASL     (SP)            ;;   ROOM FOR NEW ONE.
12168  061462  005266  000002       17$: INC     2(SP)           ;;KEEP COUNT OF CHAR
12169  061466  056616  177776            BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
12170  061472  000667                     BR      7$              ;;GET THE NEXT ONE
12171  061474  104401  001216       18$: TYPE    ,$QUES          ;;TYPE ?<CR><LF>
12172  061500  000720                     BR      20$             ;;SIMULATE CONTROL-U
12173                           .DSABL  LSB
12174
12175
```

```
12176                                   ;;*****************************************************************
12177                                   ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
12178                                   ;*CALL:
12179                                   ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
12180                                   ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
12181                                   ;*                              ;;WITH PARITY BIT STRIPPED OFF
12182                                   ;
12183
12184  061502  011646                   $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
12185  061504  016666  000004  000002           MOV     4(SP),2(SP)     ;;THE PS
12186  061512  005066  000004                   CLR     4(SP)           ;;GET READY FOR A CHARACTER
12187  061516  005046                           CLR     -(SP)           ;;PUT NEW PS ON STACK
12188  061520  012746  061526                   MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
12189  061524  000002                           RTI                     ;;POP NEW PC AND PS
12190  061526                           64$:
12191  061526  005737  060642           1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
12192  061532  001775                           BEQ     1$
12193  061534  005337  060642                   DEC     $TKCNT          ;;DECREMENT THE COUNTER
12194  061540  117766  177102  000004           MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
12195  061546  005237  060646                   INC     $TKQOUT         ;;UPDATE THE POINTER
12196  061552  023727  060646  060651           CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
12197  061560  001003                           BNE     2$              ;;BRANCH IF NO
12198  061562  012737  060650  060646           MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
12199  061570  000002                   2$:     RTI                     ;;RETURN
12200                                   ;;*****************************************************************
12201                                   ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
12202                                   ;*CALL:
12203                                   ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
12204                                   ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
12205                                   ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
12206
12207  061572  010346                   $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
12208  061574  005046                           CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
12209  061576  012703  062026           1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
12210  061602  022703  062036           2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
12211  061606  101456                           BLOS    4$              ;;BR IF YES
12212  061610  104411                           RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
12213  061612  112613                           MOVB    (SP)+,(R3)      ;;GET CHARACTER
12214  061614  122713  000177           10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
12215  061620  001022                           BNE     5$              ;;BR IF NO
12216  061622  005716                           TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
12217  061624  001007                           BNE     6$              ;;BR IF NO
12218  061626  112737  000134  062024           MOVB    #'\,9$          ;;TYPE A BACK SLASH
12219  061634  104401  062024                   TYPE    ,9$
12220  061640  012716  177777                   MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
12221  061644  005303                   6$:     DEC     R3              ;;BACKUP BY ONE
12222  061646  020327  062036                   CMP     R3,#$TTYIN      ;;STACK EMPTY?
12223  061652  103434                           BLO     4$              ;;BR IF YES
12224  061654  111337  062024                   MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
12225  061660  104401  062024                   TYPE    ,9$             ;;GO TYPE
12226  061664  000746                           BR      2$              ;;GO READ ANOTHER CHAR.
12227  061666  005716                   5$:     TST     (SP)            ;;RUBOUT KEY SET?
12228  061670  001406                           BEQ     7$              ;;BR IF NO
12229  061672  112737  000134  062024           MOVB    #'\,9$          ;;TYPE A BACK SLASH
12230  061700  104401  062024                   TYPE    ,9$
12231  061704  005016                           CLR     (SP)            ;;CLEAR THE RUBOUT KEY
```

```
12232  061706  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
12233  061712  001003                          BNE     8$              ;;BR IF NO
12234  061714  104401  062043                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
12235  061720  000726                          BR      1$              ;;GO START OVER
12236  061722  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
12237  061726  001011                          BNE     3$              ;;BRANCH IF NO
12238  061730  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
12239  061732  104401  001217                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
12240  061736  104401  062026                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
12241  061742  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
12242  061744  104401  001216          4$:     TYPE    ,$QUES          ;;TYPE A '?'
12243  061750  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
12244  061752  111337  062024          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
12245  061756  104401  062024                  TYPE    ,9$
12246  061762  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
12247  061766  001305                          BNE     2$              ;;LOOP IF NOT RETURN
12248  061770  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
12249  061774  104401  001220                  TYPE    ,$LF            ;;TYPE A LINE FEED
12250  062000  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
12251  062002  012603                          MOV     (SP)+,R3        ;;RESTORE R3
12252  062004  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
12253  062006  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
12254  062014  012766  062026  000004          MOV     #$TTYIN,4(SP)
12255  062022  000002                          RTI                     ;;RETURN
12256  062024     000          9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
12257  062025     000                  .BYTE   0                       ;;TERMINATOR
12258  062026  000010          $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
12259  062036  041536  005015     000  $CNTLC: .ASCIZ  /↑C/<15><12>    ;;CONTROL "C"
12260  062043     136  006525  000012  $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
12261  062050  043536  005015     000  $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
12262  062055     015  051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
12263  062062  036440  000040
12264  062066  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
12265  062074  020075     000
12266          062100                  .EVEN
12267                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
12268
12269                                  ;;*****************************************************************
12270                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
12271                                  ;*CHANGE IT TO BINARY.
12272                                  ;*CALL:
12273                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
12274                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
12275                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
12276
12277  062100  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
12278  062102  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
12279  062110  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12280  062112  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12281  062114  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12282  062116  104412                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
12283  062120  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
12284  062122  005001                          CLR     R1              ;;CLEAR DATA WORD
12285  062124  005002                          CLR     R2
12286  062126  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
12287  062130  001412                          BEQ     3$              ;;IF ZERO GET OUT
```

```
12288  062132  006301                  ASL     R1              ;;*2
12289  062134  006102                  ROL     R2
12290  062136  006301                  ASL     R1              ;;*4
12291  062140  006102                  ROL     R2
12292  062142  006301                  ASL     R1              ;;*8
12293  062144  006102                  ROL     R2
12294  062146  042716  177770          BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
12295  062152  062601                  ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
12296  062154  000764                  BR      2$              ;;LOOP
12297  062156  005726          3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
12298  062160  010166  000012          MOV     R1,12(SP)       ;;SAVE THE RESULT
12299  062164  010237  062200          MOV     R2,$HIOCT
12300  062170  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
12301  062172  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
12302  062174  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
12303  062176  000002                  RTI                     ;;RETURN
12304  062200  000000          $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
12305                           .SBTTL  TRAP DECODER
12306
12307                           ;;**************************************************************
12308                           ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
12309                           ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
12310                           ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
12311                           ;*GO TO THAT ROUTINE.
12312
12313  062202  016646  000002  $TRAP:  MOV     2(SP),-(SP)     ;;ASSUME THE STATUS OF
12314  062206  042716  000020          BIC     #20,(SP)        ;; THE CALLER--DO NOT ALLOW
12315  062212  012746  062220          MOV     #1$,-(SP)       ;; T-BIT TRAPS
12316  062216  000002                  RTI                     ;;SET THE NEW STATUS
12317  062220  010046          1$:     MOV     R0,-(SP)        ;;SAVE R0
12318  062222  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
12319  062226  005740                  TST     -(R0)           ;;BACKUP BY 2
12320  062230  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
12321  062232  006300                  ASL     R0              ;;POSITION FOR INDEXING
12322  062234  016000  062254          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
12323  062240  000200                  RTS     R0              ;;GO TO ROUTINE
12324
12325
12326                           ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
12327
12328  062242  011646          $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
12329  062244  016666  000004  000002  MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
12330  062252  000002                  RTI                     ;;RESTORE THE PSW
12331
12332                           .SBTTL  TRAP TABLE
12333
12334                           ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
12335                           ;*BY THE "TRAP" INSTRUCTION.
12336
12337                           ;       ROUTINE
12338                           ;       -------
12339  062254  062242          $TRPAD: .WORD   $TRAP2
12340  062256  057550                  $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
12341  062260  057346                  $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
12342  062262  057322                  $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
12343  062264  057362                  $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
```

```
12344  062266  057076                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
12345  062270  057022                          $TYPBN  ;;CALL=TYPBN    TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER
12346
12347  062272  061230                          $GTSWR  ;;CALL=GTSWR    TRAP+7(104407)  GET SOFT-SWR SETTING
12348
12349  062274  061140                          $CKSWR  ;;CALL=CKSWR    TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
12350  062276  061502                          $RDCHR  ;;CALL=RDCHR    TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
12351  062300  061572                          $RDLIN  ;;CALL=RDLIN    TRAP+12(104412) TTY TYPEIN STRING ROUTINE
12352  062302  062100                          $RDOCT  ;;CALL=RDOCT    TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
12353  062304  056726                          $SAVREG ;;CALL=SAVREG   TRAP+14(104414) SAVE R0-R5 ROUTINE
12354  062306  056764                          $RESREG ;;CALL=RESREG   TRAP+15(104415) RESTORE R0-R5 ROUTINE
12355                                  .SBTTL  POWER DOWN AND UP ROUTINES
12356
12357                                  ;;**********************************************************************
12358                                  ;POWER DOWN ROUTINE
12359  062310  012737  062450  000024 $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
12360  062316  012737  000340  000026         MOV     #340,@#PWRVEC+2 ;;PRIO:7
12361  062324  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12362  062326  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12363  062330  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
12364  062332  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
12365  062334  010446                         MOV     R4,-(SP)        ;;PUSH R4 ON STACK
12366  062336  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
12367  062340  017746  116610                 MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
12368  062344  010637  062454                 MOV     SP,$SAVR6       ;;SAVE SP
12369  062350  012737  062362  000024         MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
12370  062356  000000                         HALT
12371  062360  000776                         BR      .-2             ;;HANG UP
12372
12373                                  ;;**********************************************************************
12374                                  ;POWER UP ROUTINE
12375  062362  012737  062450  000024 $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
12376  062370  013706  062454                 MOV     $SAVR6,SP       ;;GET SP
12377  062374  005037  062454                 CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
12378  062400  005237  062454         1$:     INC     $SAVR6          ;;WAIT FOR THE INC
12379  062404  001375                         BNE     1$              ;;OF  WORD
12380  062406  012677  116542                 MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
12381  062412  012605                         MOV     (SP)+,R5        ;;POP STACK INTO R5
12382  062414  012604                         MOV     (SP)+,R4        ;;POP STACK INTO R4
12383  062416  012603                         MOV     (SP)+,R3        ;;POP STACK INTO R3
12384  062420  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
12385  062422  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
12386  062424  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
12387  062426  012737  062310  000024         MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
12388  062434  012737  000340  000026         MOV     #340,@#PWRVEC+2 ;;PRIO:7
12389  062442  104401                         TYPE                    ;REPORT THE POWER FAILURE
12390  062444  062456                 $PWRMG: .WORD   $POWER           ;;POWER FAIL MESSAGE POINTER
12391  062446  000002                         RTI
12392  062450  000000                 $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
12393  062452  000776                         BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
12394  062454  000000                 $SAVR6: 0                       ;;PUT THE SP HERE
12395  062456  005015  047520  042527 $POWER: .ASCIZ  <15><12>"POWER"
12396  062464  000122
12397                                          .EVEN
12398                                  .SBTTL  APT COMMUNICATIONS ROUTINE
12399
```

```
12400                                   ;;*****************************************************************
12401  062466  112737  000001  062732  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
12402  062474  112737  000001  062730  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
12403  062502  000403                  BR      $ATYC
12404  062504  112737  000001  062732  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
12405  062512                          $ATYC:
12406  062512  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
12407  062514  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
12408  062516  105737  062730                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
12409  062522  001450                          BEQ     5$              ;;IF NOT:  BR
12410  062524  122737  000001  001242        CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
12411  062532  001031                          BNE     3$              ;;IF NOT:  BR
12412  062534  132737  000100  001243        BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
12413  062542  001425                          BEQ     3$              ;;IF NOT:  BR
12414  062544  017600  000004                MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
12415  062550  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
12416  062556  005737  001222          1$:   TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
12417  062562  001375                          BNE     1$              ;;IF NOT:  WAIT
12418  062564  010037  001236                MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
12419  062570  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
12420  062572  001376                          BNE     2$
12421  062574  163700  001236                SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
12422  062600  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
12423  062602  010037  001240                MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
12424  062606  012737  000004  001222        MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
12425  062614  000413                          BR      5$
12426  062616  017637  000004  062642  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
12427  062624  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
12428  062632  013746  177776                MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
12429  062636  004737  057550                JSR     PC,$TYPE        ;;CALL TYPE MACRO
12430  062642  000000                  4$:     .WORD   0
12431  062644                          5$:
12432  062644  105737  062732          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
12433  062650  001416                          BEQ     12$             ;;IF NOT:  BR
12434  062652  005737  001242                TST     $ENV            ;;RUNNING UNDER APT?
12435  062656  001413                          BEQ     12$             ;;IF NOT:  BR
12436  062660  005737  001222          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
12437  062664  001375                          BNE     11$             ;;IF NOT:  WAIT
12438  062666  017637  000004  001224        MOV     @4(SP),$FATAL   ;;GET ERROR #
12439  062674  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
12440  062702  005237  001222                INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
12441  062706  105037  062732          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
12442  062712  105037  062731                  CLRB    $LFLG           ;;CLEAR LOG FLAG
12443  062716  105037  062730                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
12444  062722  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
12445  062724  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
12446  062726  000207                          RTS     PC              ;;RETURN
12447  062730     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
12448  062731     000          $LFLG:  .BYTE   0               ;;LOG FLAG
12449  062732     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
12450          062734                  .EVEN
12451          000200          APTSIZE=200
12452          000001          APTENV=001
12453          000100          APTSPOOL=100
12454          000040          APTCSUP=040
12455
```

```
 12456                                  .NLIST  BEX
```

```
                            .SBTTL  CONSOLE MESSAGES

062734                      SCTMSG:
062734  005015  040503  047116  .ASCII  <CR><LF>@CANNOT RECOVER THE BAD SECTOR FILES FROM LAST @<CR><LF>
063016  051124  041501  020113  .ASCIZ  @TRACK FOR THIS DEVICE@
063044     051              CLSPRN: .ASCII  @)@
063045     075     000      EQUALS: .ASCIZ  @=@
063047     015  025012     000 PROMPT: .ASCIZ  <CR><LF>@*@
063053     077     000      QSTMRK: .ASCIZ  @?@
063055                      HELPQST:
063055     015  052012  050131  .ASCIZ  <CR><LF>@TYPE HELP TEXT (Y OR N)??@
063111                      UBUSQST:
063111     015  041412  040510  .ASCIZ  <CR><LF>@CHANGE RM03 UNIBUS ADDRESS OR VECTOR ADDRESS (Y OR N) ??@
063204  005015  051525  020105  CNSL00: .ASCIZ  <CR><LF>@USE SAME DEVICES (Y OR N) ??@
063243     015  051012  030115  CNSL01: .ASCIZ  <CR><LF>@RM03 BUS ADDRESS (@
063270  005015  047105  051124  CNSL02: .ASCII  <CR><LF>@ENTRY NOT IN I/O PAGE@
063317     015  040412  042104          .ASCIZ  <CR><LF>@ADDRESS MUST BE >160000@
063351     015  051012  030115  CNSL03: .ASCIZ  <CR><LF>@RM03 VECTOR ADDRESS (@
063401     015  042412  052116  CNSL04: .ASCII  <CR><LF>@ENTRY OUT OF RANGE@
063425     015  040412  042104          .ASCIZ  <CR><LF>@ADDRESS MUST BE <1000@
063455     015  051012  030115  CNSL05: .ASCIZ  <CR><LF>@RM03 INTERRUPT PRIORITY (@
063511     015  042412  052116  CNSL06: .ASCIZ  <CR><LF>@ENTRY OUT OF RANGE@
063536                      CNSL07:
063536  005015  054524  042520  .ASCII  <CR><LF>@TYPE (A) TO TEST ALL DEVICES, OR TYPE DEVICE@
063614  047040  046525  042502  .ASCII  @ NUMBER(S)@
063626  005015  042524  046522  .ASCIZ  <CR><LF>@TERMINATE INPUT WITH CARRIAGE RETURN@
        063676                  .EVEN
```

```
                        .SBTTL  FUNCTION CODE TABLE

                        ;THE FUNCTION CODE TABLE IS USED TO DEFINE STATUS CONDITIONS FOR
                        ;EACH FUNCTION CODE.  BIT USAGE IS AS FOLLOWS:

                        ;       ATA - BIT 15 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF ATA SHOULD BE SET WHEN THE FUNCTION CODE IS EXECUTED, OTHERWISE,
                        ;BIT 15 IS ZERO, INDICATING THAT ATA SHOULD NOT NORMALLY BE SET.
                        ;NOTE THAT ATA MAY BE SET WHEN A COMMAND IS EXECUTED EVEN THOUGH
                        ;IT IS NOT EXPECTED AS A RESULT OF THE COMMAND.

                        ;       WCE - BIT 14 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF WRITE CHECK ERRORS ARE ENABLED AS A FUNCTION OF THE COMMAND.

                        ;       OPI - BIT 13 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF OPI ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.

                        ;       IVC - BIT 12 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF IVC ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.

                        ;       WLE - BIT 11 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF WRITE ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.
                        ;THE WRITE ERRORS WHICH ARE ENABLED ARE "WLE", "WCF", "DPE", "UPE".

                        ;       IAE - BIT 10 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF INVALID ADDRESS ERROR IS ENABLED FOR THAT COMMAND.

                        ;       AOE - BIT 09 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF READ AND WRITE ERRORS ARE ENABLED DURING THE EXECUTION OF THE
                        ;COMMAND.  THE ERRORS ENABLED BY THIS BIT ARE "TRE", "DLT", "NEM",
                        ;"MXF", "LBT", AND "AOE".

                        ;               BIT 08 IS NOT USED.

                        ;       HCE - BIT 07 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF HEADER ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.
                        ;HEADER ERRORS INCLUDE "HCRC", "HCE", "FER", AND "BSE".

                        ;       ECH - BIT 06 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF DATA FIELD ERRORS ARE ENABLED DURING THE EXECUTION OF THAT
                        ;COMMAND.  THESE ERRORS INCLUDE "MDPE", "DCK", AND "ECH".

                        ;               BIT 05 IS NOT USED.

                        ;               BIT 04 IS NOT USED.

                        ;               BIT 03 IS NOT USED.

                        ;               BIT 02 IS NOT USED.

                        ;               BIT 01 IS NOT USED.

                        ;       ILF - BIT 00 IS SET IF THE FUNCTION CODE IS ILLEGAL.

063676                  FNCDTB:                                 ;FUNCTION CODE TABLE
```

FUNCTION CODE TABLE

```
063676  020000          .WORD   OPI                                     ;NOP
063700  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (2)
063702  132000          .WORD   ATA!OPI!IVC!IAE                         ;SEEK
063704  130000          .WORD   ATA!OPI!IVC                             ;RECALIBRATE
063706  020000          .WORD   OPI                                     ;DRIVE CLEAR
063710  030000          .WORD   OPI!IVC                                 ;RELEASE
063712  130000          .WORD   OPI!ATA!IVC                             ;OFFSET
063714  130000          .WORD   OPI!ATA!IVC                             ;RETURN TO CENTERLINE
063716  020000          .WORD   OPI                                     ;READ IN PRESET
063720  020000          .WORD   OPI                                     ;PACK ACKNOWLEDGE
063722  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (24)
063724  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (26)
063726  132000          .WORD   ATA!OPI!IVC!IAE                         ;SEARCH
063730  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (32)
063732  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (34)
063734  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (36)
063736  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (40)
063740  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (42)
063742  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (44)
063744  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (46)
063746  073300          .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH             ;WRITE CHECK DATA
063750  073300          .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH             ;WRITE CHECK HEADER AND DATA
063752  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (54)
063754  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (56)
063756  037200          .WORD   OPI!IVC!WLE!IAE!AOE!HCE                 ;WRITE DATA
063760  037000          .WORD   OPI!IVC!WLE!IAE!AOE                     ;WRITE HEADER AND DATA
063762  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (64)
063764  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (66)
063766  033300          .WORD   OPI!IVC!IAE!AOE!HCE!ECH                 ;READ DATA
063770  033300          .WORD   OPI!IVC!IAE!AOE!HCE!ECH                 ;READ HEADER AND DATA
063772  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (74)
063774  130001          .WORD   OPI!ATA!ILF!IVC                         ;ILLEGAL FUNCTION (76)
```

```
                        .SBTTL  ATTENTION (ATA) TABLE

063776  001             ATNTBL: .BYTE   1.
063777  002                     .BYTE   2.
064000  004                     .BYTE   4.
064001  010                     .BYTE   8.
064002  020                     .BYTE   16.
064003  040                     .BYTE   32.
064004  100                     .BYTE   64.
064005  200                     .BYTE   128.
```

```
                        .SBTTL  DATA PATTERN TABLE

064006                  RGDTPT:
064006                  MIXED:
064006  000000                  .WORD   0.
064010  000001                  .WORD   1.
064012  000003                  .WORD   3.
064014  000007                  .WORD   7.
064016  000017                  .WORD   15.
064020  000037                  .WORD   31.
064022  000077                  .WORD   63.
064024  000177                  .WORD   127.
064026  000377                  .WORD   255.
064030  000777                  .WORD   511.
064032  001777                  .WORD   1023.
064034  003777                  .WORD   2047.
064036  007777                  .WORD   4095.
064040  017777                  .WORD   8191.
064042  037777                  .WORD   16383.
064044  077777                  .WORD   32767.
064046  177777          ONES:   .WORD   65535.
064050  177777                  .WORD   65535.
064052  077777                  .WORD   32767.
064054  037777                  .WORD   16383.
064056  017777                  .WORD   8191.
064060  007777                  .WORD   4095.
064062  003777                  .WORD   2047.
064064  001777                  .WORD   1023.
064066  000777                  .WORD   511.
064070  000377                  .WORD   255.
064072  000177                  .WORD   127.
064074  000077                  .WORD   63.
064076  000037                  .WORD   31.
064100  000017                  .WORD   15.
064102  000007                  .WORD   7.
064104  000003                  .WORD   3.
064106  000001                  .WORD   1.
064110  000000          ZEROS:  .WORD   0.
064112  000000                  .WORD   0.
064114  000001                  .WORD   1.
064116  000002                  .WORD   2.
064120  000004                  .WORD   4.
064122  000010                  .WORD   8.
064124  000020                  .WORD   16.
064126  000040                  .WORD   32.
064130  000100                  .WORD   64.
064132  000200                  .WORD   128.
064134  000400                  .WORD   256.
064136  001000                  .WORD   512.
064140  002000                  .WORD   1024.
064142  004000                  .WORD   2048.
064144  010000                  .WORD   4096.
064146  020000                  .WORD   8192.
064150  040000                  .WORD   16384.
064152  100000                  .WORD   32768.
064154  100000                  .WORD   32768.
```

```
064156  040000                  .WORD   16384.
064160  020000                  .WORD   8192.
064162  010000                  .WORD   4096.
064164  004000                  .WORD   2048.
064166  002000                  .WORD   1024.
064170  001000                  .WORD   512.
064172  000400                  .WORD   256.
064174  000200                  .WORD   128.
064176  000100                  .WORD   64.
064200  000040                  .WORD   32.
064202  000020                  .WORD   16.
064204  000010                  .WORD   8.
064206  000004                  .WORD   4.
064210  000002                  .WORD   2.
064212  000001                  .WORD   1.
064214  000000                  .WORD   0.
064216  177777                  .WORD   65535.
064220  177776                  .WORD   65534.
064222  177774                  .WORD   65532.
064224  177770                  .WORD   65528.
064226  177760                  .WORD   65520.
064230  177740                  .WORD   65504.
064232  177700                  .WORD   65472.
064234  177600                  .WORD   65408.
064236  177400                  .WORD   65280.
064240  177000                  .WORD   65024.
064242  176000                  .WORD   64512.
064244  174000                  .WORD   63488.
064246  170000                  .WORD   61440.
064250  160000                  .WORD   57344.
064252  140000                  .WORD   49152.
064254  100000                  .WORD   32768.
064256  000000                  .WORD   0.
064260  000000                  .WORD   0.
064262  100000                  .WORD   32768.
064264  140000                  .WORD   49152.
064266  160000                  .WORD   57344.
064270  170000                  .WORD   61440.
064272  174000                  .WORD   63488.
064274  176000                  .WORD   64512.
064276  177000                  .WORD   65024.
064300  177400                  .WORD   65280.
064302  177600                  .WORD   65408.
064304  177700                  .WORD   65472.
064306  177740                  .WORD   65504.
064310  177760                  .WORD   65520.
064312  177770                  .WORD   65528.
064314  177774                  .WORD   65532.
064316  177776                  .WORD   65534.
064320  177777                  .WORD   65535.
064322  125252          EARLY:  .WORD   43690.
064324  152525                  .WORD   43690./2
064326  125252                  .WORD   43690.
064330  177777                  .WORD   65535.
064332  177776                  .WORD   65534.
064334  177775                  .WORD   65533.
```

```
064336  177773          .WORD   65531.
064340  177767          .WORD   65527.
064342  177757          .WORD   65519.
064344  177737          .WORD   65503.
064346  177677          .WORD   65471.
064350  177577          .WORD   65407.
064352  177377          .WORD   65279.
064354  176777          .WORD   65023.
064356  175777          .WORD   64511.
064360  173777          .WORD   63487.
064362  167777          .WORD   61439.
064364  157777          .WORD   57343.
064366  137777          .WORD   49151.
064370  077777          .WORD   32767.
064372  077777          .WORD   32767.
064374  137777          .WORD   49151.
064376  157777          .WORD   57343.
064400  167777          .WORD   61439.
064402  173777          .WORD   63487.
064404  175777          .WORD   64511.
064406  176777          .WORD   65023.
064410  177377          .WORD   65279.
064412  177577          .WORD   65407.
064414  177677          .WORD   65471.
064416  177737          .WORD   65503.
064420  177757          .WORD   65519.
064422  177767          .WORD   65527.
064424  177773          .WORD   65531.
064426  177775          .WORD   65533.
064430  177776          .WORD   65534.
064432  177777          .WORD   65535.
064434                  ENRGDT:
```

```
                                    .SBTTL  ERROR MESSAGE TABLE

064434  071030  000000          EMT1:   .WORD   EMS1,0
064440  071077  071114  000000  EMT2:   .WORD   EMS2,EMS3,0
064446  071077  071157  000000  EMT3:   .WORD   EMS2,EMS4,0
064454  071222  071252  000000  EMT4:   .WORD   EMS5,EMS6,0
064462  071222  071364  000000  EMT5:   .WORD   EMS5,EMS10,0
064470  076057  073245  000000  EMT6:   .WORD   EMS167,EMS64,0
064476  074013  076104  000000  EMT7:   .WORD   EMS110,EMS170,0
064504  071317  000000          EMT10:  .WORD   EMS7,0
064510  071364  000000          EMT11:  .WORD   EMS10,0
064514  071426  071437  000000  EMT12:  .WORD   EMS11,EMS12,0
064522  071500  071511  071522  EMT13:  .WORD   EMS13,EMS14,EMS15,EMS16,0
064534  071574  073245  000000  EMT14:  .WORD   EMS17,EMS64,0
064542  071426  071657  000000  EMT15:  .WORD   EMS11,EMS21,0
064550  071426  071702  072031  EMT16:  .WORD   EMS11,EMS22,EMS27,0
064560  071426  071716  072042  EMT17:  .WORD   EMS11,EMS23,EMS30,0
064570  071426  071744  072042  EMT20:  .WORD   EMS11,EMS24,EMS30,0
064600  071426  071773  072031  EMT21:  .WORD   EMS11,EMS25,EMS27,0
064610  071426  072010  072031  EMT22:  .WORD   EMS11,EMS26,EMS27,0
064620  071426  072052  072042  EMT23:  .WORD   EMS11,EMS31,EMS30,0
064630  071426  072101  072042  EMT24:  .WORD   EMS11,EMS32,EMS30,0
064640  071426  072130  072042  EMT25:  .WORD   EMS11,EMS33,EMS30,0
064650  071426  072156  072042  EMT26:  .WORD   EMS11,EMS34,EMS30,0
064660  071426  072227  072042  EMT27:  .WORD   EMS11,EMS35,EMS30,0
064670  071426  072256  072042  EMT30:  .WORD   EMS11,EMS36,EMS30,0
064700  071426  072305  072042  EMT31:  .WORD   EMS11,EMS37,EMS30,0
064710  071426  072333  072042  EMT32:  .WORD   EMS11,EMS40,EMS30,0
064720  071426  072362  072042  EMT33:  .WORD   EMS11,EMS41,EMS30,0
064730  071426  072410  072042  EMT34:  .WORD   EMS11,EMS42,EMS30,0
064740  071426  072437  072042  EMT35:  .WORD   EMS11,EMS43,EMS30,0
064750  071426  072466  072042  EMT36:  .WORD   EMS11,EMS44,EMS30,0
064760  071426  072541  072042  EMT37:  .WORD   EMS11,EMS45,EMS30,0
064770  073331  071634  000000  EMT40:  .WORD   EMS66,EMS20,0
064776  073517  075225  073525  EMT41:  .WORD   EMS75,EMS141,EMS76,0
065006  075316  075326  073453  EMT42:  .WORD   EMS144,EMS145,EMS72,EMS76,0
065020  072633  072751  073525  EMT43:  .WORD   EMS47,EMS53,EMS76,0
065030  073556  072751  073525  EMT44:  .WORD   EMS77,EMS53,EMS76,0
065040  073604  072751  073525  EMT45:  .WORD   EMS100,EMS53,EMS76,0
065050  073632  072751  073525  EMT46:  .WORD   EMS101,EMS53,EMS76,0
065060  073263  073245  000000  EMT47:  .WORD   EMS65,EMS64,0
065066  071500  071522  073217  EMT50:  .WORD   EMS13,EMS15,EMS63,0
065076  071426  073604  072042  EMT51:  .WORD   EMS11,EMS46,EMS30,EMS67,0
065110  072633  072751  073401  EMT52:  .WORD   EMS47,EMS53,EMS67,EMS115,EMS140,EMS141,0
065126  072633  072751  073401  EMT53:  .WORD   EMS47,EMS53,EMS67,EMS115,EMS141,EMS164,0
065144  072662  072751  073401  EMT54:  .WORD   EMS50,EMS53,EMS67,0
065154  075353  075270  072751  EMT55:  .WORD   EMS142,EMS143,EMS53,EMS67,0
065166  072711  073453  073401  EMT56:  .WORD   EMS51,EMS72,EMS67,EMS115,EMS50,EMS70,0
065204  076057  073463  073401  EMT57:  .WORD   EMS167,EMS73,EMS67,0
065214  073034  073401  074213  EMT60:  .WORD   EMS56,EMS67,EMS115,EMS150,EMS152,EMS70,0
065232  073440  073034  073401  EMT61:  .WORD   EMS71,EMS56,EMS67,EMS115,EMS150,EMS152,EMS72,0
065252  075204  073401  074213  EMT62:  .WORD   EMS140,EMS67,EMS115,EMS47,EMS70,0
065266  000000                  EMT63:  .WORD
065270  075411  075454  073426  EMT64:  .WORD   EMS150,EMS152,EMS70,EMS115,EMS56,EMS73,EMS67,0
065310  072737  076504  076665  EMT65:  .WORD   EMS52,EMS205,EMS214,EMS206,EMS115,EMS51,EMS72,0
065330  076016  073707  073453  EMT66:  .WORD   EMS165,EMS103,EMS72,EMS124,0
```

```
065342  076016  073707  073453  EMT67:  .WORD   EMS165,EMS103,EMS72,EMS171,0
065354  072604  071634  075711  EMT70:  .WORD   EMS46,EMS20,EMS163,0
065364  073440  073632  075711  EMT71:  .WORD   EMS71,EMS101,EMS163,0
065374  072633  073453  075711  EMT72:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS140,EMS141,0
065412  072633  073453  075711  EMT73:  .WORD   EMS47,EMS72,EMS163,EMS115,EMS141,EMS72,0
065430  073034  072751  075711  EMT74:  .WORD   EMS56,EMS53,EMS163,0
065440  073440  073034  075711  EMT75:  .WORD   EMS71,EMS56,EMS163,EMS115,EMS150,EMS152,EMS72,0
065460  072662  072751  075711  EMT76:  .WORD   EMS50,EMS53,EMS163,0
065470  075253  075270  072751  EMT77:  .WORD   EMS142,EMS143,EMS53,EMS163,0
065502  073517  075225  075711  EMT100: .WORD   EMS75,EMS141,EMS163,EMS115,EMS47,EMS70,0
065520  073517  075411  075711  EMT101: .WORD   EMS75,EMS150,EMS163,EMS115,EMS56,EMS73,0
065536  076057  073463  075711  EMT102: .WORD   EMS167,EMS73,EMS163,0
065546  075374  075430  073500  EMT103: .WORD   EMS147,EMS151,EMS74,EMS163,0
065560  072711  073500  075711  EMT104: .WORD   EMS51,EMS74,EMS163,EMS115,EMS50,EMS70,0
065576  076016  073604  075711  EMT105: .WORD   EMS165,EMS100,EMS163,0
065606  076016  073556  075711  EMT106: .WORD   EMS165,EMS77,EMS163,0
065616  075316  075326  072751  EMT107: .WORD   EMS144,EMS145,EMS53,EMS163,EMS115,EMS143,EMS70,0
065636  074013  074053  074220  EMT110: .WORD   EMS110,EMS112,EMS116,EMS111,0
065650  074137  071157  000000  EMT111: .WORD   EMS113,EMS4,0
065656  074137  071114  000000  EMT112: .WORD   EMS113,EMS3,0
065664  074013  074213  074220  EMT113: .WORD   EMS110,EMS115,EMS116,EMS117,EMS114,0
065700  074137  074272  000000  EMT114: .WORD   EMS113,EMS120,0
065706  074307  074747  074773  EMT115: .WORD   EMS121,EMS132,EMS133,0
065716  074352  074747  074773  EMT116: .WORD   EMS122,EMS132,EMS133,0
065726  074407  074747  074773  EMT117: .WORD   EMS123,EMS132,EMS133,0
065736  074452  074747  074773  EMT120: .WORD   EMS124,EMS132,EMS133,
065746  074504  074747  074773  EMT121: .WORD   EMS125,EMS132,EMS133,0
065756  074547  074747  074773  EMT122: .WORD   EMS126,EMS132,EMS133,0
065766  075147  074747  074773  EMT123: .WORD   EMS137,EMS132,EMS133,0
065776  074651  074747  074773  EMT124: .WORD   EMS130,EMS132,EMS133,0
066006  074707  074747  074773  EMT125: .WORD   EMS131,EMS132,EMS133,0
066016  074307  074747  075016  EMT126: .WORD   EMS121,EMS132,EMS134,EMS123,0
066030  074352  074747  075016  EMT127: .WORD   EMS122,EMS132,EMS134,EMS123,0
066042  074407  074747  075016  EMT130: .WORD   EMS123,EMS132,EMS134,EMS123,0
066054  074452  074747  075016  EMT131: .WORD   EMS124,EMS132,EMS134,EMS123,0
066066  074504  074747  075016  EMT132: .WORD   EMS125,EMS132,EMS134,EMS123,0
066100  074547  074747  075016  EMT133: .WORD   EMS126,EMS132,EMS134,EMS123,0
066112  075147  074747  075016  EMT134: .WORD   EMS137,EMS132,EMS134,EMS123,0
066124  074651  074747  075016  EMT135: .WORD   EMS130,EMS132,EMS134,EMS123,0
066136  074707  074747  075016  EMT136: .WORD   EMS131,EMS132,EMS134,EMS123,0
066150  074307  074747  075060  EMT137: .WORD   EMS121,EMS132,EMS135,0
066160  074407  074747  075060  EMT140: .WORD   EMS123,EMS132,EMS135,0
066170  074307  074747  075122  EMT141: .WORD   EMS121,EMS132,EMS136,0
066200  075147  074747  075122  EMT142: .WORD   EMS137,EMS132,EMS136,0
066210  074452  074747  075122  EMT143: .WORD   EMS124,EMS132,EMS136,0
066220  074504  074747  075122  EMT144: .WORD   EMS125,EMS132,EMS136,0
066230  074547  074747  075122  EMT145: .WORD   EMS126,EMS132,EMS136,0
066240  074707  074747  075122  EMT146: .WORD   EMS131,EMS132,EMS136,0
066250  075656  074747  075122  EMT147: .WORD   EMS162,EMS132,EMS136,0
066260  074651  074747  075122  EMT150: .WORD   EMS130,EMS132,EMS136,0
066270  075204  074213  075225  EMT151: .WORD   EMS140,EMS115,EMS141,EMS70,0
066302  075253  074213  075270  EMT152: .WORD   EMS142,EMS115,EMS143,EMS72,0
066314  075316  075326  075355  EMT153: .WORD   EMS144,EMS145,EMS146,EMS115,EMS143,EMS72,0
066332  075316  075326  071634  EMT154: .WORD   EMS144,EMS145,EMS20,EMS115,EMS143,EMS70,0
066350  075411  075454  075522  EMT155: .WORD   EMS150,EMS152,EMS154,EMS153,0
066362  075374  075430  075522  EMT156: .WORD   EMS147,EMS151,EMS154,EMS155,0
```

```
066374  075374  075430  075556  EMT157: .WORD   EMS147,EMS151,EMS156,EMS157,0
066406  075626  075556  075611  EMT160: .WORD   EMS161,EMS156,EMS160,0
066416  071773  072031  075556  EMT161: .WORD   EMS25,EMS27,EMS156,EMS160,0
066430  072010  072031  075556  EMT162: .WORD   EMS26,EMS27,EMS156,EMS160,0
066442  072633  075711  075204  EMT163: .WORD   EMS47,EMS163,EMS140,0
066452  072633  071634  000000  EMT164: .WORD   EMS47,EMS20,0
066460  073034  071634  000000  EMT165: .WORD   EMS56,EMS20,0
066466  072604  071634  000000  EMT166: .WORD   EMS46,EMS20,0
066474  077205  071634  000000  EMT167: .WORD   EMS229,EMS20,0
066502  076057  075522  076032  EMT170: .WORD   EMS167,EMS154,EMS166,0
066512  076057  075522  075574  EMT171: .WORD   EMS167,EMS154,EMS157,0
066522  076057  075522  075536  EMT172: .WORD   EMS167,EMS154,EMS155,0
066532  076133  074747  074773  EMT173: .WORD   EMS171,EMS132,EMS133,0
066542  076133  074747  075016  EMT174: .WORD   EMS171,EMS132,EMS134,EMS123,0
066554  074137  076306  000000  EMT175: .WORD   EMS113,EMS177,0
066562  076330  076345  000000  EMT176: .WORD   EMS200,EMS201,0
066570  076443  075225  072042  EMT177: .WORD   EMS203,EMS141,EMS30,EMS202,0
066602  076443  072711  072042  EMT200: .WORD   EMS203,EMS51,EMS30,EMS202,0
066614  076443  076456  072042  EMT201: .WORD   EMS203,EMS204,EMS30,EMS202,0
066626  076443  072662  072042  EMT202: .WORD   EMS203,EMS50,EMS30,EMS202,0
066640  076443  075270  072042  EMT203: .WORD   EMS203,EMS143,EMS30,EMS202,0
066652  075411  073500  076413  EMT204: .WORD   EMS150,EMS74,EMS202,EMS115,EMS152,EMS72,0
066670  072662  073707  073453  EMT205: .WORD   EMS50,EMS103,EMS72,0
066700  073716  073463  076413  EMT206: .WORD   EMS104,EMS73,EMS202,0
066710  073517  075225  074213  EMT207: .WORD   EMS75,EMS141,EMS115,EMS140,0
066722  073517  075411  000000  EMT210: .WORD   EMS75,EMS150,0
066730  072711  073453  074213  EMT211: .WORD   EMS51,EMS72,EMS115,EMS50,EMS70,0
066744  075253  072751  074213  EMT212: .WORD   EMS142,EMS53,EMS115,EMS143,EMS72,0
066760  072662  073707  072751  EMT213: .WORD   EMS50,EMS103,EMS53,0
066770  072737  076504  076032  EMT214: .WORD   EMS52,EMS205,EMS166,EMS206,EMS115,EMS51,EMS72,0
067010  072737  074250  073034  EMT215: .WORD   EMS52,EMS117,EMS56,EMS163,0
067022  073034  072042  073245  EMT216: .WORD   EMS56,EMS30,EMS64,0
067032  072604  072042  073245  EMT217: .WORD   EMS46,EMS30,EMS64,0
067042  072052  072042  073245  EMT220: .WORD   EMS31,EMS30,EMS64,0
067052  072633  072042  073245  EMT221: .WORD   EMS47,EMS30,EMS64,0
067062  072737  074250  073556  EMT222: .WORD   EMS52,EMS117,EMS77,0
067072  072737  074250  073217  EMT223: .WORD   EMS52,EMS117,EMS63,0
067102  000000                  EMT224: .WORD
067104  000000                  EMT225: .WORD
067106  000000                  EMT226: .WORD
067110  000000                  EMT227: .WORD
067112  000000                  EMT230: .WORD
067114  000000                  EMT231: .WORD
067116  000000                  EMT232: .WORD
067120  000000                  EMT233: .WORD
067122  000000                  EMT234: .WORD
067124  000000                  EMT235: .WORD
067126  000000                  EMT236: .WORD
067130  000000                  EMT237: .WORD
067132  000000                  EMT240: .WORD
067134  000000                  EMT241: .WORD
067136  000000                  EMT242: .WORD
067140  000000                  EMT243: .WORD
067142  000000                  EMT244: .WORD
067144  000000                  EMT245: .WORD
067146  076057  074747  076543  EMT246: .WORD   EMS167,EMS132,EMS207,0
```

```
067156  076057  074747  076570  EMT247: .WORD   EMS167,EMS132,EMS210,EMS125,0
067170  076057  076601  076570  EMT250: .WORD   EMS167,EMS211,EMS210,EMS207,EMS206,0
067204  076617  076642  000000  EMT251: .WORD   EMS212,EMS213,0
067212  076016  076617  000000  EMT252: .WORD   EMS165,EMS212,0
067220  075374  075430  075556  EMT253: .WORD   EMS147,EMS151,EMS156,EMS210,EMS26,EMS27,0
067236  076443  073632  072042  EMT254: .WORD   EMS203,EMS101,EMS30,0
067246  076443  076057  072042  EMT255: .WORD   EMS203,EMS167,EMS30,0
067256  076443  073604  072042  EMT256: .WORD   EMS203,EMS100,EMS30,0
067266  071426  072604  072042  EMT257: .WORD   EMS11,EMS46,EMS30,EMS102,0
067300  072737  074250  073034  EMT260: .WORD   EMS52,EMS117,EMS56,EMS102,0
067312  072737  076504  077007  EMT261: .WORD   EMS52,EMS205,EMS130,EMS206,EMS115,EMS51,EMS72,0
067332  073716  073463  076413  EMT262: .WORD   EMS104,EMS73,EMS202,0
067342  072662  072751  073660  EMT263: .WORD   EMS50,EMS53,EMS102,0
067352  073034  072751  073160  EMT264: .WORD   EMS56,EMS53,EMS102,EMS115,EMS150,EMS152,EMS70,0
067372  073440  073034  073160  EMT265: .WORD   EMS71,EMS56,EMS102,EMS115,EMS150,EMS152,EMS72,0
067412  075253  075270  072751  EMT266: .WORD   EMS142,EMS143,EMS53,EMS102,0
067424  072662  075355  074213  EMT267: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,0
067442  072633  072751  073660  EMT270: .WORD   EMS47,EMS53,EMS102,EMS115,EMS140,0
067456  072633  072751  073660  EMT271: .WORD   EMS47,EMS53,EMS102,EMS115,EMS141,EMS72,0
067474  073517  075225  073660  EMT272: .WORD   EMS75,EMS141,EMS102,EMS115,EMS47,EMS73,0
067512  072711  073500  073660  EMT273: .WORD   EMS51,EMS74,EMS102,EMS115,EMS50,EMS70,0
067530  073217  072751  073111  EMT274: .WORD   EMS63,EMS53,EMS57,EMS115,EMS41,EMS146,0
067546  074220  072751  072362  EMT275: .WORD   EMS116,EMS53,EMS41,EMS57,0
067560  072633  072751  073111  EMT276: .WORD   EMS47,EMS53,EMS57,EMS115,EMS140,0
067574  072633  072751  073111  EMT277: .WORD   EMS47,EMS53,EMS57,EMS115,EMS141,EMS72,0
067612  073034  072751  073111  EMT300: .WORD   EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS70,0
067632  073440  073034  072751  EMT301: .WORD   EMS71,EMS56,EMS53,EMS57,EMS115,EMS150,EMS152,EMS72,0
067654  076016  073604  073707  EMT302: .WORD   EMS165,EMS100,EMS103,EMS57,0
067666  076016  073632  073707  EMT303: .WORD   EMS165,EMS101,EMS103,EMS57,0
067700  076016  073556  073707  EMT304: .WORD   EMS165,EMS77,EMS103,EMS57,0
067712  072604  072042  073245  EMT305: .WORD   EMS46,EMS30,EMS64,EMS57,0
067724  072662  072751  073111  EMT306: .WORD   EMS50,EMS53,EMS57,0
067734  072662  075355  074213  EMT307: .WORD   EMS50,EMS146,EMS115,EMS52,EMS117,EMS46,EMS57,0
067754  075253  075270  072751  EMT310: .WORD   EMS142,EMS143,EMS53,EMS57,0
067766  073716  073707  072751  EMT311: .WORD   EMS104,EMS103,EMS53,EMS57,0
070000  073745  073707  072751  EMT312: .WORD   EMS105,EMS103,EMS53,EMS57,0
070012  075316  075326  073707  EMT313: .WORD   EMS144,EMS145,EMS103,EMS57,EMS115,EMS143,EMS70,0
070032  072410  073707  072751  EMT314: .WORD   EMS42,EMS103,EMS53,EMS57,0
070044  072052  073707  072751  EMT315: .WORD   EMS31,EMS103,EMS53,EMS57,0
070056  073440  072052  073707  EMT316: .WORD   EMS71,EMS31,EMS103,EMS57,0
070070  072437  073707  073111  EMT317: .WORD   EMS43,EMS103,EMS57,0
070100  072541  073707  073111  EMT320: .WORD   EMS45,EMS103,EMS57,0
070110  072466  073707  073111  EMT321: .WORD   EMS44,EMS103,EMS57,0
070120  073774  071634  000000  EMT322: .WORD   EMS106,EMS30,0
070126  072256  073707  073111  EMT323: .WORD   EMS36,EMS103,EMS57,0
070136  076206  072256  073707  EMT324: .WORD   EMS173,EMS36,EMS103,EMS57,0
070150  076166  072256  073707  EMT325: .WORD   EMS172,EMS36,EMS103,EMS57,0
070162  071500  076223  071522  EMT326: .WORD   EMS13,EMS174,EMS15,EMS35,EMS53,EMS175,0
070200  075374  075430  073500  EMT327: .WORD   EMS147,EMS151,EMS74,EMS175,0
070212  073331  072751  076231  EMT330: .WORD   EMS66,EMS53,EMS175,0
070222  072130  073707  072751  EMT331: .WORD   EMS33,EMS103,EMS53,EMS175,0
070234  072333  073707  072751  EMT332: .WORD   EMS40,EMS103,EMS53,EMS57,0
070246  072711  073500  073111  EMT333: .WORD   EMS51,EMS74,EMS57,EMS115,EMS50,EMS70,0
070264  073517  075225  073111  EMT334: .WORD   EMS75,EMS141,EMS57,EMS115,EMS47,EMS73,0
070302  073517  075411  075454  EMT335: .WORD   EMS75,EMS150,EMS152,EMS57,EMS115,EMS56,EMS73,0
070322  073137  073152  073201  EMT336: .WORD   EMS60,EMS61,EMS62,0
```

```
070332  074220  074250  072156  EMT337: .WORD   EMS116,EMS117,EMS34,0
070342  072156  072751  072763  EMT340: .WORD   EMS34,EMS53,EMS54,EMS111,0
070354  073005  072156  000000  EMT341: .WORD   EMS55,EMS34,0
070362  072737  074250  073034  EMT342: .WORD   EMS52,EMS117,EMS56,EMS57,0
070374  072737  074250  072541  EMT343: .WORD   EMS52,EMS117,EMS45,EMS57,0
070406  072737  074250  072466  EMT344: .WORD   EMS52,EMS117,EMS44,EMS57,0
070420  072737  074250  077027  EMT345: .WORD   EMS52,EMS117,EMS221,0
070430  073774  071634  074213  EMT346: .WORD   EMS106,EMS20,EMS115,EMS223,EMS72,0
070444  072737  076504  077077  EMT347: .WORD   EMS52,EMS205,EMS222,EMS206,0
070456  073517  075411  073660  EMT350: .WORD   EMS75,EMS150,EMS102,EMS115,EMS56,EMS73,0
070474  076057  073463  073660  EMT351: .WORD   EMS167,EMS73,EMS102,0
070504  076^03  000000          EMT352: .WORD   EMS215,0
070510  076754  076443  076724  EMT353: .WORD   EMS217,EMS203,EMS216,0
070520  077027  071634  000000  EMT354: .WORD   EMS221,EMS20,0
```

```
070526  077257  100073  100154  EHT1:   .WORD   EH1,STSH1,STSH2,STSH4,0
070540  100073  100154  100307  EHT2:   .WORD   STSH1,STSH2,STSH4,0

070550  077276  000000          EHT110: .WORD   EH110,0
070554  077305  000000          EHT111: .WORD   EH111,0

070560  077324  000000          EHT114: .WORD   EH114,0
070564  077353  100073  100154  EHT223: .WORD   EH223,STSH1,STSH2,STSH4,0
070576  077401  100073  100154  EHT256: .WORD   EH256,STSH1,STSH2,STSH4,0

070610  077456  100073  100154  EHT336: .WORD   EH336,STSH1,STSH2,STSH4,0
070622  077515  100073  100154  EHT337: .WORD   EH337,STSH1,STSH2,STSH4,0
070634  077654  100073  100154  EHT344: .WORD   EH344,STSH1,STSH2,STSH4,0

070646  100014  000000          EHT353: .WORD   EH353,0
```

```
070652  100350  100444  100462  EDT1:    .WORD   ED1,STSD1,STSD2,STSD4
070662  100444  100462  100514  EDT2:    .WORD   STSD1,STSD2,STSD4

070670  100356                  EDT110:  .WORD   ED110
070672  100362                  EDT111:  .WORD   ED111

070674  100370                  EDT114:  .WORD   ED114
070676  100400  100444  100462  EDT223:  .WORD   ED223,STSD1,STSD2,STSD4

070706  100410  100444  100462  EDT336:  .WORD   ED336,STSD1,STSD2,STSD4
070716  100422  100444  100462  EDT337:  .WORD   ED337,STSD1,STSD2,STSD4
070726  100422  100444  100462  EDT344:  .WORD   ED337,STSD1,STSD2,STSD4,0

070740  100434                  EDT353:  .WORD   ED353
```

ERROR MESSAGE TABLE

```
070742  100527  100545  100545  EFT1:    .WORD   EF111,STSF,STSF,STSF
070752  100545  100545  100545  EFT2:    .WORD   STSF,STSF,STSF

070760  100526                  EFT110:  .WORD   EF110
070762  100527                  EFT111:  .WORD   EF111

070764  100531                  EFT114:  .WORD   EF114
070766  100531  100545  100545  EFT223:  .WORD   EF114,STSF,STSF,STSF

070776  100534  100545  100545  EFT336:  .WORD   EF336,STSF,STSF,STSF
071006  100534  100545  100545  EFT337:  .WORD   EF336,STSF,STSF,STSF
071016  100534  100545  100545  EFT344:  .WORD   EF336,STSF,STSF,STSF

071026  100531                  EFT353:  .WORD   EF114
```

```
                                .SBTTL  ERROR MESSAGE STRINGS

071030 051127 047117 020107 EMS1:   .ASCIZ  @WRONG UNIT SELECTED (RMCS2, BITS 0-2) @
071077    104 053105 041511 EMS2:   .ASCIZ  @DEVICE HENT @
071114 047125 053101 044501 EMS3:   .ASCIZ  @UNAVAILABLE "DVA" (RMCS1, BIT 11) @
071157    116 047117 054105 EMS4:   .ASCIZ  @NONEXISTENT "NED" (RMCS2, BIT 12) @
071222 047503 046515 047101 EMS5:   .ASCIZ  @COMMAND NOT COMPLETED, @
071252 047503 052116 047522 EMS6:   .ASCIZ  @CONTROLLER NOT READY (RMCS1, BIT 7) @
071317    104 044522 042526 EMS7:   .ASCIZ  @DRIVE NOT READY "DRY" (RMDS, BIT 7) @
071364 047507 047040 052117 EMS10:  .ASCIZ  @GO NOT RESET "GO" (RMCS1, BIT 0) @
071426 047111 040526 044514 EMS11:  .ASCIZ  @INVALID @
071437    106 047125 052103 EMS12:  .ASCIZ  @FUNCTION CODE (RMCS1, BITS 1-5) @
071500 040515 051523 052502 EMS13:  .ASCIZ  @[illegible]BUS @
071511    103 047117 051124 EMS14:  .ASCIZ  @CONTROL @
071522 052502 020123 040520 EMS15:  .ASCIZ  @BUS PARITY ERROR @
071544 046442 050103 021105 EMS16:  .ASCIZ  @"MCPE" (RMCS1, BIT 13) @
071574 051124 047101 043123 EMS17:  .ASCIZ  @TRANSFER ERROR (RMCS1, BIT 14) @
071634 044123 052517 042114 EMS20:  .ASCIZ  @SHOULD NOT BE SET @
071657    127 051117 020104 EMS21:  .ASCIZ  @WORD COUNT (RMWC) @
071702 052502 020123 051050 EMS22:  .ASCIZ  @[illegible]US (RMBA) @
071716 046042 052102 020042 EMS23:  .ASCIZ  @"LBT" (RMDS, BIT 10) @
071744 040442 042517 020042 EMS24:  .ASCIZ  @"AOE" (RMER1, BIT 09) @
071773    104 051511 020113 EMS25:  .ASCIZ  @DISK (RMDA) @
072010 054503 044514 042116 EMS26:  .ASCIZ  @CYLINDER (RMDC) @
072031    101 042104 042522 EMS27:  .ASCIZ  @ADDRESS @
072042 052123 052101 051525 EMS30:  .ASCIZ  @STATUS @
072052 053442 042514 020042 EMS31:  .ASCIZ  @"WLE" (RMER1, BIT 11) @
072101    042 050125 021105 EMS32:  .ASCIZ  @"UPE" (RMCS2, BIT [illegible]) @
072130 053442 043103 020042 EMS33:  .ASCIZ  @"WCF" (RMER1, BIT 5) @
072156 051127 052111 020105 EMS34:  .ASCIZ  @WRITE CHECK ERROR-"WCE" (RMCS2, BIT 14) @
072227    042 042115 042520 EMS35:  .ASCIZ  @"MDPE" (RMCS2, BIT 8) @
072256 042042 045503 020042 EMS36:  .ASCIZ  @"DCK" (RMER1, BIT 15) @
072305    042 041505 021110 EMS37:  .ASCIZ  @"ECH" (RMER1, BIT 6) @
072333    042 046104 021124 EMS40:  .ASCIZ  @"DLT" (RMCS2, BIT 15) @
072362 046442 043130 020042 EMS41:  .ASCIZ  @"MXF" (RMCS2, BIT 9) @
072410 042042 042524 020042 EMS42:  .ASCIZ  @"DTE" (RMER1, BIT 12) @
072437    042 041510 041522 EMS43:  .ASCIZ  @"HCRC" (RMER1, BIT 8) @
072466 042510 042101 051105 EMS44:  .ASCIZ  @HEADER COMPARE ERROR "HCE" (RMER1, BIT 7) @
072541    106 051117 040515 EMS45:  .ASCIZ  @FORMAT ERROR "FER" (RMER1, BIT 4) @
072604 044442 042501 020042 EMS46:  .ASCIZ  @"IAE" (RMER1, BIT 10) @
072633    042 050117 021111 EMS47:  .ASCIZ  @"OPI" (RMER1, BIT 13) @
072662 051442 044513 020042 EMS50:  .ASCIZ  @"SKI" (RMER2, BIT 14) @
072711    042 044520 021120 EMS51:  .ASCIZ  @"PIP" (RMDS, BIT 13) @
072737    124 042510 051040 EMS52:  .ASCIZ  @THE RM03 @
072751    104 052105 041505 EMS53:  .ASCIZ  @DETECTED @
072763    101 020124 047101 EMS54:  .ASCIZ  @AT AN UNEXPECTED @
073005    111 041516 051117 EMS55:  .ASCIZ  @INCORRECT DATA DURING @
073034 047111 040526 044514 EMS56:  .ASCIZ  @INVALID COMMAND ERROR "IVC" (RMER2, BIT 12) @
073111    104 051125 047111 EMS57:  .ASCIZ  @DURING DATA TRANSFER @
073137    104 052101 020101 EMS60:  .ASCIZ  @DATA READ @
073152 047504 051505 047040 EMS61:  .ASCIZ  @DOES NOT COMPARE WITH @
073201    104 052101 020101 EMS62:  .ASCIZ  @DATA WRITTEN @
073217    042 040520 021122 EMS63:  .ASCIZ  @"PAR" (RMER1, BIT 3) @
073245    111 020123 047111 EMS64:  .ASCIZ  @IS INCORRECT @
073263    103 046517 047520 EMS65:  .ASCIZ  @COMPOSITE ERROR "ERR" (RMDS, BIT 14) @
073331    104 052101 020101 EMS66:  .ASCIZ  @DATA PARITY ERROR "DPE" (RMER2, BIT 3) @
```

```
073401      104  051125  047111  EMS67:  .ASCIZ  @DURING SEEK COMMAND @
073426  051511  051040  051505  EMS70:  .ASCIZ  @IS RESET @
073440  051105  047522  042516  EMS71:  .ASCIZ  @ERRONEOUS @
073453      111  020123  042523  EMS72:  .ASCIZ  @IS SET @
073463      104  042111  047040  EMS73:  .ASCIZ  @DID NOT SET @
073500  044504  020104  047516  EMS74:  .ASCIZ  @DID NOT RESET @
073517      114  051517  020124  EMS75:  .ASCIZ  @LOST @
073535      104  051125  047111  EMS76:  .ASCIZ  @DURING PACK ACK COMMAND @
073556  051042  051115  020042  EMS77:  .ASCIZ  @"RMR" (RMER1, BIT 2) @
073604  044442  051114  020042  EMS100: .ASCIZ  @"ILR" (RMER1, BIT 1) @
073632  044442  043114  020042  EMS101: .ASCIZ  @"ILF" (RMER1, BIT 0) @
073660  052504  044522  043516  EMS102: .ASCIZ  @DURING SEARCH COMMAND @
073707      105  051122  051117  EMS103: .ASCIZ  @ERROR @
073716  046042  041502  020042  EMS104: .ASCIZ  @"LBC" (RMER2, BIT 10) @
073745      042  051514  021103  EMS105: .ASCIZ  @"LSC" (RMER2, BIT 11) @
073774  042510  042101  051105  EMS106: .ASCIZ  @HEADER ERRORS @
074013      102  051525  052040  EMS110: .ASCIZ  @BUS TIMEOUT (04 TRAP) @
074042  042101  051104  051505  EMS111: .ASCIZ  @ADDRESS @
074053      127  042510  020116  EMS112: .ASCII  @WHEN READING/WRITING RH REGISTERS @
074115      101  020124  044124          .ASCIZ  @AT THE FOLLOWING @
074137      124  042510  051440  EMS113: .ASCIZ  @THE SELECTED DEVICE IS @
074167      116  047117  054105  EMS114: .ASCIZ  @NONEXISTENT DEVICE @
074213      040  006455  000012  EMS115: .ASCIZ  @ -@<CR><LF>
074220  044124  020105  040515  EMS116: .ASCIZ  @THE MASSBUS CONTROLLER @
074250  040506  046111  042105  EMS117: .ASCIZ  @FAILED TO DETECT @
074272  047516  020124  047101  EMS120: .ASCIZ  @NOT AN RM03 @
074307      103  047117  051124  EMS121: .ASCIZ  @CONTROL STATUS REGISTER 1, RMCS1, @
074352  052502  020123  042101  EMS122: .ASCIZ  @BUS ADDRESS REGISTER, RMBA, @
074407      103  047117  051124  EMS123: .ASCIZ  @CONTROL STATUS REGISTER 2, RMCS2, @
074452  051105  047522  020122  EMS124: .ASCIZ  @ERROR REGISTER 1, RMER1, @
074504  052101  042524  052116  EMS125: .ASCIZ  @ATTENTION SUMMARY REGISTER, RMAS, @
074547      115  044501  052116  EMS126: .ASCIZ  @MAINTENANCE REGISTER #1, RMMR #1, @
074612  041505  020103  047520  EMS127: .ASCIZ  @ECC POSITION REGISTER, RMEC1, @
074651      105  041503  050040  EMS130: .ASCIZ  @ECC PATTERN REGISTER, RMEC2, @
074707      115  044501  052116  EMS131: .ASCIZ  @MAINTENANCE REGISTER 2, RMMR2, @
074747      116  052117  044440  EMS132: .ASCIZ  @NOT INITIALIZED BY @
074773      125  044516  052502  EMS133: .ASCIZ  @UNIBUS INITIALIZE @
075016  047503  052116  047522  EMS134: .ASCIZ  @CONTROLLER CLEAR, I.E.  BIT 5 OF @
075060  044122  030461  042440  EMS135: .ASCIZ  @PH11 ERROR CLEAR (RMCS1, BIT 14) @
075122  051104  053111  020105  EMS136: .ASCIZ  @DRIVE CLEAR COMMAND @
075147      104  044522  042526  EMS137: .ASCIZ  @DRIVE STATUS REGISTER, RMDS @
075204  042515  044504  046525  EMS140: .ASCIZ  @MEDIUM OFF LINE @
075225      042  047515  021114  EMS141: .ASCIZ  @"MOL" (RMDS, BIT 12) @
075253      104  044522  042526  EMS142: .ASCIZ  @DRIVE FAULT @
075270  042042  041526  020042  EMS143: .ASCIZ  @"DVC" (RMER2, BIT 7) @
075316  047125  040523  042506  EMS144: .ASCIZ  @UNSAFE @
075326  052442  051516  020042  EMS145: .ASCIZ  @"UNS" (RMER1, BIT 14) @
075355      123  047510  046125  EMS146: .ASCIZ  @SHOULD BE SET @
075374  043117  051506  052105  EMS147: .ASCIZ  @OFFSET MODE @
075411      040  047526  052514  EMS150: .ASCIZ   @ VOLUME VALID @
075430  047442  021115  024040  EMS151: .ASCIZ  @"OM" (RMDS, BIT 0) @
075454  053042  021126  024040  EMS152: .ASCIZ  @"VV" (RMDS, BIT 6) @
075500  040520  045503  040440  EMS153: .ASCIZ  @PACK ACK COMMAND @
075522  047516  020124  042523  EMS154: .ASCIZ  @NOT SET BY @
075536  043117  051506  052105  EMS155: .ASCIZ  @OFFSET COMMAND @
075556  047516  020124  042522  EMS156: .ASCIZ  @NOT RESET BY @
```

```
075574 052122 020103 047503 EMS157: .ASCIZ  @BIC COMMAND @
075611    122 050111 041440 EMS160: .ASCIZ  @RIP COMMAND @
075626 043117 051506 052105 EMS161: .ASCIZ  @OFFSET REGISTER (RMOF) @
075656 051105 047522 020122 EMS162: .ASCIZ  @ERROR REGISTER #2, RMER2, @
075711    104 051125 047111 EMS163: .ASCIZ  @DURING RECALIBRATE @
075735    111 020123 047111 EMS164: .ASCII  @IS INTERMITTENT OR DRIVE DIDNT DROP ON @
076004 054503 044514 042116         .ASCIZ  @CYLINDER @
076016 047125 054105 042520 EMS165: .ASCIZ  @UNEXPECTED @
076032 042522 040503 044514 EMS166: .ASCIZ  @RECALIBRATE COMMAND @
076057    042 052101 021101 EMS167: .ASCIZ  @"ATA" (RMDS, BIT15) @
076104 044127 047105 051040 EMS170: .ASCIZ  @WHEN READING REGISTER @
076133    105 051122 051117 EMS171: .ASCIZ  @ERROR REGISTER #2, RMER2, @
076166 047516 051116 041505 EMS172: .ASCIZ  @NONRECOVERABLE @
076206 042522 047503 042526 EMS173: .ASCIZ  @RECOVERABLE @
076223    104 052101 020101 EMS174: .ASCIZ  @DATA @
076231    104 051125 047111 EMS175: .ASCIZ  @DURING WRITE COMMAND @
076257    042 050117 020105 EMS176: .ASCIZ  @"OPE" (RMER2, BIT 13) @
076306 047111 053440 044522 EMS177: .ASCIZ  @IN WRITE PROTECT @
076330 040503 020116 047516 EMS200: .ASCIZ  @CAN NOT SET @
076345    104 040511 047107 EMS201: .ASCIZ  @DIAGNOSTIC MODE "DMD" (RMMR1, BIT 0) @
076413    104 051125 047111 EMS202: .ASCIZ  @DURING DIAGNOSTIC MODE @
076443    111 041516 051117 EMS203: .ASCIZ  @INCORRECT @
076456 053442 046122 020042 EMS204: .ASCIZ  @"WRL" (RMDS, BIT 11) @
076504 054105 041505 052125 EMS205: .ASCIZ  @EXECUTED @
076516 044527 044124 041440 EMS206: .ASCIZ  @WITH COMP ERROR SET @
076543    042 047507 020042 EMS207: .ASCIZ  @"GO" (RMCS1, BIT 0) @
076570 051127 052111 047111 EMS210: .ASCIZ  @WRITING @
076601    127 051501 051040 EMS211: .ASCIZ  @WAS RESET BY @
076617    120 047522 051107 EMS212: .ASCIZ  @PROGRAM INTERRUPT @
076642 040527 020123 047516 EMS213: .ASCIZ  @WAS NOT GENERATED @
076665    123 042505 020113 EMS214: .ASCIZ  @SEEK COMMAND @
076703    120 047522 051107 EMS215: .ASCIZ  @PROGRAM TIMEOUT @
076724 052504 044522 043516 EMS216: .ASCIZ  @DURING LOOK AHEAD TEST @
076754 047514 045517 040440 EMS217: .ASCIZ  @LOOK AHEAD REGISTER,RMLA, @
077007    123 040505 041522 EMS220: .ASCIZ  @SEARCH COMMAND @
077027    102 042101 051440 EMS221: .ASCIZ  @BAD SECTOR ERROR "BSE" (RMER2, BIT 15) @
077077    101 042040 052101 EMS222: .ASCIZ  @A DATA TRANSFER COMMAND @
077130 042510 042101 051105 EMS223: .ASCIZ  @HEADER COMPARE INHIBIT "HCI" (RMOF, BIT 10) @
077203    116 047117 054105 EMS224: .ASCIZ  @NONEXISTENT MEMORY "NEM" (RMCS2, BIT 11) @
```

```
077257    105  050130  052103  EH1:    .ASCIZ  @EXPCTD  RECEVD@
077276 052502  040523  051104  EH110:  .ASCIZ  @BUSADR@
077305    040  046522  051503  EH111:  .ASCIZ  @ RMCS2   RMCS1@

077324 042522  042503  042126  EH114:  .ASCIZ  @RECEVD  SNGPRT  DULPRT@
077353    105  050130  052103  EH223:  .ASCIZ  @EXPCTD  RECEVD   DATA@
077401    105  050130  052103  EH256:  .ASCII  @EXPCTD  RECEVD  RGSTR@<CR><LF>
077430 052123  052101  051525          .ASCIZ  @STATUS  STATUS  INDEX@

077456 042107  042101  051522  EH336:  .ASCIZ  @GDADRS  GDDATA  BDADRS  BDDATA@
077515    122  041515  031123  EH337:  .ASCII  @RMCS2   STATUS  FAILING    DATA@<CR><LF>
077555    137  057537  057537          .ASCII  @+++++++++++++++  +++++++++++++++@<CR><LF>
077615    105  050130  052103          .ASCIZ  @EXPCTD  RECEVD    BIT    ADRESS@

077654 046522  051105  020061  EH344:  .ASCII  @RMER1   STATUS  HEADER  FAILING@<CR><LF>
077715    137  057537  057537          .ASCII  @+++++++++++++++   WORD      BIT@<CR><LF>
077754 054105  041520  042124          .ASCIZ  @EXPCTD  RECEVD  NUMBER  POSITON@
100014 054105  041520  042124  EH353:  .ASCII  @EXPCTD  RECEVD@<CR><LF>
100034 041474  037122  046074          .ASCIZ  @<CR><LF> RMLA    RMLA    RMOF @

100073    040  046522  051503  STSH1:  .ASCII  @ RMCS1  RMCS2   RMDS    RMER1   RMER2@<CR><LF>
100142 020040  051040  040515          .ASCIZ  @   RMAS@<CR><LF>
100154 051040  053515  020103  STSH2:  .ASCII  @ RMWC    RMBA    RMDA    RMOF    RMDC@<CR><LF>
100223    040  020040  051040          .ASCIZ  @     RMEC1   RMEC2@<CR><LF>
100247    040  046522  040504  STSH3:  .ASCIZ  @ RMDA    RMDC    RMOF    RMLA@<CR><LF>
100307    040  046522  051115  STSH4:  .ASCIZ  @ RMMR1   RMMR2   RMDT    RMSN@<CR><LF>
```

```
                100350                          .EVEN
        100350  001140  001142  000000  ED1:    .WORD   $GDDAT,$BDDAT,0
        100356  001276  000000          ED110:  .WORD   $BASE,0
        100362  001174  001176  000000  ED111:  .WORD   $TMP0,$TMP1,0

        100370  001354  001176  001200  ED114:  .WORD   RMDTI,$TMP1,$TMP2,0
        100400  001140  001142  001174  ED223:  .WORD   $GDDAT,$BDDAT,$TMP0,0

        100410  001134  001140  001136  ED336:  .WORD   $GDADR,$GDDAT,$BDADR,$BDDAT,0

        100422  001140  001142  001174  ED337:  .WORD   $GDDAT,$BDDAT,$TMP0,$TMP1,0
        100434  001140  001142  001430  ED353:  .WORD   $GDDAT,$BDDAT,RMOF0,0

        100444  001326  001336  001340  STSD1:  .WORD   RMCS1I,RMCS2I,RMDSI,RMER1I,RMER2I,RMASI,0
        100462  001330  001332  001334  STSD2:  .WORD   RMWCI,RMBAI,RMDAI,RMOFI,RMDCI,RMEC1I
        100476  001374  000000          .WORD   RMEC2I,0
        100502  001334  001362  001360  STSD3:  .WORD   RMDAI,RMDCI,RMOFI,RMLAI,0
        100514  001352  001366  001354  STSD4:  .WORD   RMMR1I,RMMR2I,RMDTI,RMSNI,0
```

```
100526  000             EF110:  .BYTE   0
100527  000  000        EF111:  .BYTE   0,0
100531  000  000  000   EF114:  .BYTE   0,0,0
100534  000  000  000   EF336:  .BYTE   0,0,0,0
100540  000  000  000   EF337:  .BYTE   0,0,0,0,0

100545  000  000  000   STSF:   .BYTE   0,0,0,0,0,0,0
```

```
100554  000402
101560  177777
101562  177777
101564  000402
102570  177777
102572  177777

                .EVEN
MFGFIL: .BLKW   258.
                .WORD   -1
                .WORD   -1
USRFIL:         .BLKW   258.
                .WORD   -1
                .WORD   -1
.EVEN
```

```
                                        BUFFER:
102574                                  BUFONE: .BLKW   258.
102574  000402                          BUFTWO: .BLKW   258.
103600  000402


        102574                          .       =       BUFFER
102574                                  HELP:
102574  005015                          .ASCII  <CR><LF>
102576  040503  052125  047511          .ASCII  @CAUTION: PARTS 2 AND 3 OF THE FUNCTIONAL TEST LEAVE @<CR><LF>
102664  040502  020104  042510          .ASCII  @BAD HEADERS ON THE PACK SO BE SURE TU FORMAT THE PACKS@<CR><LF>
102754  044127  047105  042040          .ASCII  @WHEN DONE.@<CR><LF>
102770  046011  051511  020124          .ASCII  @       LIST OF TESTS@<CR><LF>
103010  057411  057537  057537          .ASCII  @       ++++++++++++++@<CR><LF>
103030  030524  041411  047117          .ASCII  @T1     CONTROLLER ACCESS TEST@<CR><LF>
103063     124  004462  042504          .ASCII  @T2     DEVICE AVAILABLE TEST@<CR><LF>
103115     124  004463  051104          .ASCII  @T3     DRIVE TYPE TEST@<CR><LF>
103141     124  004464  047506          .ASCII  @T4     FORMAT ZEROS - 18@<CR><LF>
103167     124  004465  047506          .ASCII  @T5     FORMAT ZEROS - 16@<CR><LF>
103215     124  004466  042532          .ASCII  @T6     ZERO FILL TEST@<CR><LF>
103240  033524  043011  051117          .ASCII  @T7     FORMAT CHECK ZEROS - 16@<CR><LF>
103274  030524  004460  047506          .ASCII  @T10    FORMAT CHECK ZEROS W/ WCE ERROR@<CR><LF>
103341     124  030461  043011          .ASCII  @T11    FORMAT ONES - 16@<CR><LF>
103367     124  031061  043011          .ASCII  @T12    FORMAT CHECK ONES - 16@<CR><LF>
103423     124  031461  043011          .ASCII  @T13    FORMAT CHECK ONES W/ WCE ERRORS@<CR><LF>
103470  030524  004464  047506          .ASCII  @T14    FORMAT MULTIPLE SECTORS@<CR><LF>
103525     124  032461  043011          .ASCII  @T15    FORMAT W/ HEAD SWITCHING@<CR><LF>
103563     124  033061  043011          .ASCII  @T16    FORMAT W/ MID TRANSFER SEEK@<CR><LF>
103624  030524  004467  047506          .ASCII  @T17    FORMAT W/ IMPLIED SEEK@<CR><LF>
103660  031124  004460  047506          .ASCII  @T20    FORMAT EACH SECTOR ADDRESS@<CR><LF>
103720  031124  004461  047506          .ASCII  @T21    FORMAT EACH TRACK ADDRESS@<CR><LF>
103757     124  031062  043011          .ASCII  @T22    FORMAT PRIME CYLINDERS@<CR><LF>
104013     124  031462  043011          .ASCII  @T23    FORMAT LAST SECTOR@<CR><LF>
104043     124  032062  043011          .ASCII  @T24    FORMAT W/ AOE ERROR@<CR><LF>
104074  031124  004465  047506          .ASCII  @T25    FORMAT INVALID SECTOR ADDRESS@<CR><LF>
104137     124  033062  043011          .ASCII  @T26    FORMAT INVALID TRACK ADDRESS@<CR><LF>
104201     124  033462  043011          .ASCII  @T27    FORMAT INVALID CYLINDER ADDRESS@<CR><LF>
104246  031524  004460  047506          .ASCII  @T30    FORMAT AT OFFSET@<CR><LF>
104274  031524  004461  053111          .ASCII  @T31    IVC FORMAT TEST@<CR><LF>
104321     124  031063  043011          .ASCII  @T32    FORMAT ERROR TEST - 18@<CR><LF>
104355     124  031463  043011          .ASCII  @T33    FORMAT ERROR TEST - 16@<CR><LF>
104411     124  032063  043011          .ASCII  @T34    FORMAT HCE, FIRST HEADER WORD@<CR><LF>
104454  031524  004465  047506          .ASCII  @T35    FORMAT HCE, SECOND HEADER WORD@<CR><LF>
104520  005015                          .ASCII  <CR><LF>
104522  047411  042520  040522          .ASCII  @       OPERATIONAL SWITCH SETTINGS@<CR><LF>
104560  057411  057537  057537          .ASCII  @       ++++++++++++++++++++++++++@<CR><LF>
104616  005015                          .ASCII  <CR><LF>
104620  053523  052111  044103          .ASCII  @SWITCH                 USE@<CR><LF>
104636  026455  026455  026455          .ASCII  @------                 ----------@<CR><LF>
104674  020040  032461  004411          .ASCII  @  15           HALT ON ERROR@<CR><LF>
104721     040  030440  004464          .ASCII  @  14           LOOP ON TEST@<CR><LF>
104745     040  030440  004463          .ASCII  @  13           INHIBIT ERROR TYPEOUTS@<CR><LF>
105003     040  030440  004462          .ASCII  @  12           @<CR><LF>
105013     040  030440  004461          .ASCII  @  11           INHIBIT ITERATIONS@<CR><LF>
105045     040  030440  004460          .ASCII  @  10           BELL ON ERROR@<CR><LF>
105072  020(  )  034440  004411          .ASCII  @   9           LOOP ON ERROR@<CR><LF>
105117     040  020040  004470          .ASCII  @   8           LOOP ON TEST IN SWR<7:0>@<CR><LF>
```

```
105157     040  020040  004467  .ASCII  @   7          TN128@<CR><LF>
105174  020040  033040  004411  .ASCII  @   6          TN64@<CR><LF>
105210  020040  032440  004411  .ASCII  @   5          TN32@<CR><LF>
105224  020040  032040  004411  .ASCII  @   4          TN16@<CR><LF>
105240  020040  031440  004411  .ASCII  @   3          TN8@<CR><LF>
105253     040  020040  004462  .ASCII  @   2          TN4@<CR><LF>
105266  020040  030440  004411  .ASCII  @   1          TN2@<CR><LF>
105301     040  020040  004460  .ASCII  @   0          TN1@<CR><LF>
105314  005015                  .ASCII  <CR><LF>
105316  005015     000          .ASCIZ  <CR><LF>
        000001                  .END
```

```
ABASE = 176700          1259#   1367    1408
ACDW1 = 000000          1367    1410
ACDW2 = 000000          1367    1411
ACKSTS  047152          7989    8177    10232#
ACPUOP= 000000          1367    1382
ADDW0 = 000000          1367    1412
ADDW1 = 000000          1367    1413
ADDW10= 000000          1367
ADDW11= 000000          1367
ADDW12= 000000          1367
ADDW13= 000000          1367
ADDW14= 000000          1367
ADDW15= 000000          1367
ADDW2 = 000000          1367    1414
ADDW3 = 000000          1367    1415
ADDW4 = 000000          1367    1416
ADDW5 = 000000          1367    1417
ADDW6 = 000000          1367    1418
ADDW7 = 000000          1367    1419
ADDW8 = 000000          1367
ADDW9 = 000000          1367
ADEVCT= 000000          1367    1373
ADEVM = 000000          1367    1409
ADR   = 000001          3497#   3498    3539#   3540#   3541    3583#   3584#   3585    3617#   3618#   3619    3785#   3786#
                        3787    3953#   3954#   3955    4123#   4124#   4125    4283#   4284#   4285    4471#   4472#   4473
                        4639#   4640#   4641    4799#   4800#   4801    4987#   4988#   4989    5146#   5147#   5148    5305#
                        5306#   5307    5464#   5465#   5466    5629#   5630#   5631    5774#   5775#   5776    5919#   5920#
                        5921    6064#   6065#   6066    6149#   6150#   6151    6231#   6232#   6233    6378#   6379#   6380
                        6525#   6526#   6527    6671#   6672#   6673    6846#   6847#   6848    6999#   7000#   7001    7137#
                        7138#   7139    7275#   7276#   7277    7479#   7480#   7481
AENV  = 000000          1367    1378
AENVM = 000000          1367    1379
AFATAL= 000000          1367    1370
AMADR1= 000000          1367    1395
AMADR2= 000000          1367    1399
AMADR3= 000000          1367    1402
AMADR4= 000000          1367    1405
AMAMS1= 000000          1367    1389
AMAMS2= 000000          1367    1397
AMAMS3= 000000          1367    1400
AMAMS4= 000000          1367    1403
AMSCAD= 000000          1367    1375
AMSCLC= 000000          1367    1376
AMSCTY= 000000          1367    1369
AMTYP1= 000000          1367    139C
AMTYP2= 000000          1367    1398
AMTYP3= 000000          1367    1401
AMTYP4= 000000          1367    1404
AOE   = 001000          1039#   1050    9402    9417    9432    9447    9451    9698    9702    12456
APASS = 000000          1367    1372
APE   = 100000          1238#
APRIOR= 000000          1367
APTCSU= 000040         11803   12454#
APTENV= 000001         11796   11988   12410   12452#
APTSIZ= 000200          3274   12451#
APTSPO= 000100         11798   12412   12453#
```

```
ARGS  = 000004   3638#  3675#  3707#  3714#  3721#  3750#  3757#  3764#  3773#  3806#  3843#  3875#  3882#
                 3889#  3918#  3925#  3932#  3941#  3974#  4011#  4044#  4051#  4058#  4088#  4095#  4102#
                 4111#  4144#  4181#  4213#  4220#  4227#  4255#  4262#  4269#  4304#  4341#  4373#  4380#
                 4387#  4418#  4429#  4492#  4529#  4561#  4568#  4575#  4604#  4611#  4618#  4627#  4660#
                 4697#  4729#  4736#  4743#  4771#  4778#  4785#  4820#  4857#  4889#  4896#  4903#  4934#
                 4945#  5008#  5045#  5077#  5084#  5091#  5119#  5126#  5133#  5167#  5204#  5236#  5243#
                 5250#  5278#  5285#  5292#  5326#  5363#  5395#  5402#  5409#  5437#  5444#  5451#  5485#
                 5524#  5557#  5564#  5571#  5599#  5606#  5613#  5650#  5688#  5695#  5702#  5731#  5738#
                 5745#  5754#  5795#  5833#  5840#  5847#  5876#  5883#  5890#  5899#  5940#  5978#  5985#
                 5992#  6021#  6028#  6035#  6044#  6073#  6121#  6128#  6135#  6157#  6203#  6210#  6217#
                 6252#  6290#  6297#  6304#  6313#  6344#  6351#  6358#  6399#  6437#  6444#  6451#  6460#
                 6491#  6498#  6505#  6546#  6584#  6591#  6598#  6607#  6638#  6645#  6652#  6692#  6742#
                 6749#  6756#  6804#  6811#  6818#  6827#  6867#  6918#  6932#  6941#  6972#  6986#  7020#
                 7058#  7065#  7072#  7102#  7113#  7124#  7158#  7196#  7203#  7210#  7240#  7251#  7262#
                 7297#  7344#  7351#  7358#  7387#  7402#  7417#  7437#  7455#  7501#  7548#  7555#  7562#
                 7591#  7602#  7620#
ASNDA   001476   1490#  8367*  8396   8442*  8443   8445*  8446*  8447   8449*  8494
ASNDC   001474   1489#  8366*  8395   8450*  8451   8453*  8493
ASWREG= 000000   1367   1380
ATA   = 100000   1019#  9229   9230   9232   9964   9965   10000  10003  10495  10496  10535  10538  12456
ATESTN= 000000   1367   1371
ATNMSK= 000377   1056#  10692
ATNTBL  063776   3440   3458   12456#
AUNIT = 000000   1367   1374
AUSWR = 000000   1367   1381
AVECT1= 120254   1260#  1367   1406
AVECT2= 000000   1367   1407
A16   = 000400   1209#
A17   = 001000   1208#
BACK  = 000001   3638#  3643   3675#  3679   3707#  3711   3714#  3718   3721#  3725   3750#  3754   3757#
                 3761   3764#  3768   3773#  3779   3806#  3811   3843#  3847   3875#  3879   3882#  3886
                 3889#  3893   3918#  3922   3925#  3929   3932#  3936   3941#  3947   3974#  3979   4011#
                 4015   4044#  4048   4051#  4055   4058#  4062   4088#  4092   4095#  4099   4102#  4106
                 4111#  4117   4144#  4149   4181#  4185   4213#  4217   4220#  4224   4227#  4231   4255#
                 4259   4262#  4266   4269#  4273   4304#  4309   4341#  4345   4373#  4377   4380#  4384
                 4387#  4391   4418#  4422   4429#  4433   4492#  4497   4529#  4533   4561#  4565   4568#
                 4572   4575#  4579   4604#  4608   4611#  4615   4618#  4622   4627#  4633   4660#  4665
                 4697#  4701   4729#  4733   4736#  4740   4743#  4747   4771#  4775   4778#  4782   4785#
                 4789   4820#  4825   4857#  4861   4889#  4893   4896#  4900   4903#  4907   4934#  4938
                 4945#  4949   5008#  5013   5045#  5049   5077#  5081   5084#  5088   5091#  5095   5119#
                 5123   5126#  5130   5133#  5137   5167#  5172   5204#  5208   5236#  5240   5243#  5247
                 5250#  5254   5278#  5282   5285#  5289   5292#  5296   5326#  5331   5363#  5367   5395#
                 5399   5402#  5406   5409#  5413   5437#  5441   5444#  5448   5451#  5455   5485#  5490
                 5524#  5528   5557#  5561   5564#  5568   5571#  5575   5599#  5603   5606#  5610   5613#
                 5617   5650#  5655   5688#  5692   5695#  5699   5702#  5706   5731#  5735   5738#  5742
                 5745#  5749   5754#  5760   5795#  5800   5833#  5837   5840#  5844   5847#  5851   5876#
                 5880   5883#  5887   5890#  5894   5899#  5905   5940#  5945   5978#  5982   5985#  5989
                 5992#  5996   6021#  6025   6028#  6032   6035#  6039   6044#  6050   6073#  6078   6121#
                 6125   6128#  6132   6135#  6139   6157#  6162   6203#  6207   6210#  6214   6217#  6221
                 6252#  6257   6290#  6294   6297#  6301   6304#  6308   6313#  6318   6344#  6348   6351#
                 6355   6358#  6362   6399#  6404   6437#  6441   6444#  6448   6451#  6455   6460#  6465
                 6491#  6495   6498#  6502   6505#  6509   6546#  6551   6584#  6588   6591#  6595   6598#
                 6602   6607#  6612   6638#  6642   6645#  6649   6652#  6656   6692#  6697   6742#  6746
                 6749#  6753   6756#  6760   6804#  6808   6811#  6815   6818#  6822   6827#  6833   6867#
                 6872   6918#  6922   6932#  6936   6941#  6946   6972#  6976   6986#  6990   7020#  7025
                 7058#  7062   7065#  7069   7072#  7076   7102#  7106   7113#  7117   7124#  7128   7158#
```

```
                    7163    7196#   7200    7203#   7207    7210#   7214    7240#   7244    7251#   7255    7262#   7266
                    7297#   7302    7344#   7348    7351#   7355    7358#   7362    7387#   7391    7402#   7406    7417#
                    7421    7437#   7441    7455#   7459    7501#   7506    7548#   7552    7555#   7559    7562#   7566
                    7591#   7595    7602#   7606    7620#   7624
BAOSCT  034210      8083#
BAI   = 000010      1228#   9618
BB00  = 000001      1146#
BB01  = 000002      1145#
BB02  = 000004      1144#
BB03  = 000010      1143#
BB04  = 000020      1142#
BB05  = 000040      1141#
BB06  = 000100      1140#
BB07  = 000200      1139#
BB08  = 000400      1138#
BB09  = 001000      1137#
BIT0  = 000001       935#   8905
BIT00 = 000001       925#    935     961    1010    1029    1048    1086    1105    1146    1232    1248
BIT01 = 000002       924#    934     960    1009    1047    1085    1104    1145    1231    1247
BIT02 = 000004       923#    933     959    1008    1046    1084    1103    1144    1230    1246
BIT03 = 000010       922#    932     958    1007    1045    1083    1102    1143    1158    1228    1245
BIT04 = 000020       921#    931     957    1006    1044    1101    1142    1227
BIT05 = 000040       920#    930     956    1043    1082    1100    1141    1226
BIT06 = 000100       919#    929    1028    1042    1064    1081    1099    1140    1211    1225    1244
BIT07 = 000200       918#    928    1027    1041    1063    1080    1098    1122    1139    1157    1210    1224
BIT08 = 000400       917#    927    1005    1026    1040    1062    1079    1097    1138    1209    1222   11883
BIT09 = 001000       916#    926    1004    1025    1039    1061    1078    1096    1137    1208    1221   11901   11999
BIT1  = 000002       934#   8373    8531    9635
BIT10 = 002000       915#   1003    1024    1038    1060    1077    1095    1121    1136    1156    1207    1220    1243
                    11976
BIT11 = 004000       914#    955    1023    1037    1076    1094    1112    1120    1135    1155    1219    1242    8001
                    11908
BIT12 = 010000       913#   1022    1036    1075    1093    1119    1134    1154    1218    1241    7951
BIT13 = 020000       912#   1021    1035    1074    1092    1111    1133    1153    1205    1217    1240   11983
BIT14 = 040000       911#   1020    1034    1073    1091    1110    1132    1152    1163    1204    1216    1239    7916
                    11869
BIT15 = 100000       910#   1019    1033    1072    1090    1109    1131    1151    1162    1203    1215    1238    7903
                     8616   8670
BIT2  = 000004       933#   8905
BIT3  = 000010       932#   8029   11077   11164
BIT4  = 000020       931#   7979   11183
BIT5  = 000040       930#
BIT6  = 000100       929#   7928
BIT7  = 000200       928#   3309    8911
BIT8  = 000400       927#
BIT9  = 001000       926#
BOTADR  033232      7767*   7785*   7788    7803    7848#
BOTFLG  033234      7753*   7795*   7798    7801*   7849#
BPTVEC= 000014       942#
BSE   = 100000      1151#   7415    7426    8279    9522
BUFFER  102574      8098*   8347   12456#
BUFONE  102574      3630    3774    3798    3942    3966    4112    4136    4296    4484    4628    4652    4812    5000
                    5159    5318    5477    5642    5755    5787    5900    5932    6045    6244    6391    6538    6684
                    6828    6838    6859    7012    7150    7289    7306    7308*   7310*   7427    7446    7493    7510
                    7512*   7514*   7611   12456#
BUFTWO  103600      3732    3775    3900    3943    4070    4113   4396*    4443    4456    4586    4629   4912*    4958
```

```
                        4971    5713    5756    5858    5901    6003    6046    6086    6170    6323    6470    6617    6781
                        6829    6951    7084    7222    7369    7428    7447    7573    7612   12456#
CC     = 004000         1135#
CH     = 002000         1136#
CHRCNT  033235          7754*   7772*   7778*   7779    7782*   7786    7791*   7802*   7850#
CKSWR  = 104410        11867   11972   11998   12349#
CLEAN  = ****** U       7431   12456
CLKADR  001500          1491#   8867*   8869    8873*   8875    8900
CLKVCT  001502          1492#   8868*   8874*
CLR    = 000040         1226#  10061
CLRSTS  046272          7938   10097#
CLSPRN  063044          3361    3377    3398   12456#
CMNSTA  006632          3315    3447#
CMPBUF  036360          3773    3941    4111    4627    5754    5899    6044    6827    8596#
CMPERR  056416         11476#
CNSL00  063204          3325   12456#
CNSL01  063243          3358   12456#
CNSL02  063270          3367   12456#
CNSL03  063351          3372   12456#
CNSL04  063401          3383   12456#
CNSL05  063455          3393   12456#
CNSL06  063511          3404   12456#
CNSL07  063536          3417   12456#
CNTCLR  046154          3546    3590    7918   10053#
CONT   = 000100         1099#
CR     = 000015          850#   3430    7770   11555   11842   11852   12456
CRLF   = 000200          851#   3298   11813   11852
CYLMSK= 001777          1126#   8534
DBCK   = 100000         1072#
DBEN   = 040000         1073#
DBL    = 002000         1243#
DCK    = 100000         1033#   1050    8272    8604    9560    9580   11170   11173
DDISP  = 177570          857#   1343    3262
DEBL   = 020000         1074#
DEVSEL  044502          7905    9763#
DISPLA  001156          1343#   3262*   3270*  11923*  11975*
DISPRE  000174          1270#   3270
DLT    = 100000         1215#   9404   11249   11252
DMD    = 000001         1086#   1105#   6876
DPE    = 000010         1158#   8273    8279    9066    9346    9842   10381   10802   11223   11226
DPEHI  = 040000         1239#
DPELO  = 020000         1240#
DPR    = 000400         1026#  10003   10200   10201   10667
DRQ    = 004000         1112#
DRVCLR= 000010          970#   8126   10655
DRVSTS  051510         8144   10646#
DRY    = 000200         1027#   9033    9129    9132    9133   10003   10200   10201   10667
DSWR   = 177570          856#   1342    3261
DTASTS  052312          3714    3757    3882    3925    4051    4095    4220    4262    4380    4429    4568    4611    4736
                        4778    4896    4945    5084    5126    5243    5285    5402    5444    5564    5606    5695    5738
                        5840    5883    5985    6028    6128    6210    6297    6351    6444    6498    6591    6645    6749
                        6811    7065    7113    7203    7251    7351    7402    7417    7437    7555    7602    8247   10777#
DTE    = 010000         1036#   1050   11036   11054   11057
DTO    = 010000         1075#
DULPRT= 024024          1115#   3605    3608
DVA    = 004000          955#   3565    8756    8829    8994    8997    9788   10106   10654   10655
```

```
DVC    = 000200          1157#  9935  10190  10442  10480  10483  10616  10989  10993  10996  11041  11415  11420
                        11423
EARLY    064322          12456#
EBL    = 020000           1092#
ECH    = 000100           1042#  1050   8272   8606   9543   9558   9576   9582  11180  12456
ECI    = 004000           1120#  8608   9578  11177
ECRC   = 001000           1096#
EDT1     070652           1526  1533   1540   1547   1554   1561   1575   1582   1589   1596   1603   1610   1617
                          1624  1631   1638   1645   1652   1659   1666   1673   1680   1687   1694   1701   1708
                          1715  1722   1729   1736   1743   1750   1757   1764   1771   1778   1785   1792   1799
                          1806  1813   1821   1828   1835   1842   1849   1857   1865   1873   1887   1894   1901
                          1908  1915   1922   1929   1937   1944   1951   1958   1965   1972   1979   1986   1993
                          2000  2007   2014   2021   2063   2070   2077   2084   2091   2098   2105   21[illegible]   2119
                          2126  2133   2140   2147   2154   2161   2168   2175   2182   2189   2196   2203   2210
                          2217  2224   2231   2238   2245   2252   2259   2266   2273   2280   2287   2294   2301
                          2308  2315   2322   2337   2344   2351   2358   2372   2379   2386   2393   2400   2407
                          2414  2421   2428   2435   2442   2449   2456   2463   2470   2477   2484   2491   2498
                          2519  2526   2533   2540   2547   2687   2694   2701   2722   2729   2736   2750   2757
                          2764  2771   2778   2785   2792   2799   2807   2814   2822   2829   2836   2844   2852
                          2860  2869   2877   2885   2892   2899   2906   2913   2920   2927   2934   2941   2948
                          2955  2962   2969   2976   2983   2990   2997   3004   3011   3018   3025   3032   3039
                          3046  3053   3060   3067   3074   3081   3116   3123   3137   3144   3151   3158   3165
                          3186  12456#
EDT110   070670           2028  12456#
EDT111   070672           2035   2042  12456#
EDT114   070674           2056  12456#
EDT2     070662           2505   2512   2708   2715  12456#
EDT223   070676           2554   2743  12456#
EDT336   070706           2330   3088   3102   3109  12456#
EDT337   070716           3095  12456#
EDT344   070726           3130  12456#
EDT353   070740           3179  12456#
ED1      100350          12456#
ED110    100356          12456#
ED111    100362          12456#
ED114    100370          12456#
ED223    100400          12456#
ED336    100410          12456#
ED337    100422          12456#
ED353    100434          12456#
EECC   = 000020           1101#
EFT1     070742           1527  1534   1541   1548   1555   1562   1576   1583   1590   1597   1604   1611   1618
                          1625  1632   1639   1646   1653   1660   1667   1674   1681   1688   1695   1702   1709
                          1716  1723   1730   1737   1744   1751   1758   1765   1772   1779   1786   1793   1800
                          1807  1814   1822   1829   1836   1843   1850   1858   1866   1874   1888   1895   1902
                          1909  1916   1923   1930   1938   1945   1952   1959   1966   1973   1980   1987   1994
                          2001  2008   2015   2022   2064   2071   2078   2085   2092   2099   2106   2113   2120
                          2127  2134   2141   2148   2155   2162   2169   2176   2183   2190   2197   2204   2211
                          2218  2225   2232   2239   2246   2253   2260   2267   2274   2281   2288   2295   2302
                          2309  2316   2323   2338   2345   2352   2359   2373   2380   2387   2394   2401   2408
                          2415  2422   2429   2436   2443   2450   2457   2464   2471   2478   2485   2492   2499
                          2520  2527   2534   2541   2548   2688   2695   2702   2723   2730   2737   2751   2758
                          2765  2772   2779   2786   2793   2800   2808   2815   2823   2830   2837   2845   2853
                          2861  2870   2878   2886   2893   2900   2907   2914   2921   2928   2935   2942   2949
                          2956  2963   2970   2977   2984   2991   2998   3005   3012   3019   3026   3033   3040
                          3047  3054   3061   3068   3075   3082   3117   3124   3138   3145   3152   3159   3166
```

```
                    3187   12456#
                    2029   12456#
EFT110  070760      2036    2043   12456#
EFT111  070762      2057   12456#
EFT114  070764      2506    2513    2709    2716   12456#
EFT2    070752      2555    2744   12456#
EFT223  070766      2331    3089    3103    3110   12456#
EFT256  070776      3096   12456#
EFT336  071006      3131   12456#
EFT337  071016      3180   12456#
EFT344  071026     12456#
EFT353  071026     12456#
EF110   100526     12456#
EF111   100527     12456#
EF114   100531     12456#
EF336   100534     12456#
EF337   100540     12456#
EHT1    070526      1525    1532    1539    1546    1553    1560    1574    1581    1588    1595    1602    1609    1616
                    1623    1630    1637    1644    1651    1658    1665    1672    1763    1770    1777    1784    1791    1798
                    1714    1721    1728    1735    1742    1749    1756    1763    1770    1777    1784    1791    1798
                    1805    1812    1820    1827    1834    1841    1848    1856    1864    1872    1886    1893    1900
                    1907    1914    1921    1928    1936    1943    1950    1957    1964    1971    1978    1985    1992
                    1999    2006    2013    2020    2062    2069    2076    2083    2090    2097    2104    2111    2118
                    2125    2132    2139    2146    2153    2160    2167    2174    2181    2188    2195    2202    2209
                    2216    2223    2230    2237    2244    2251    2258    2265    2272    2279    2286    2293    2300
                    2307    2314    2321    2336    2343    2350    2357    2371    2378    2385    2392    2399    2406
                    2413    2420    2427    2434    2441    2448    2455    2462    2469    2476    2483    2490    2497
                    2518    2525    2532    2539    2546    2686    2693    2700    2721    2728    2735    2749    2756
                    2763    2770    2777    2784    2791    2798    2806    2813    2821    2828    2835    2843    2851
                    2859    2868    2876    2884    2891    2898    2905    2912    2919    2926    2933    2940    2947
                    2954    2961    2968    2975    2982    2989    2996    3003    3010    3017    3024    3031    3038
                    3045    3052    3059    3066    3073    3080    3115    3122    3136    3143    3150    3157    3164
                    3185   12456#
EHT110  070550      2027   12456#
EHT111  070554      2034    2041   12456#
EHT114  070560      2055   12456#
EHT2    070540      2504    2511    2707    2714   12456#
EHT223  070564      2553   12456#
EHT256  070576      2742   12456#
EHT336  070610      2329    3087    3101    3108   12456#
EHT337  070622      3094   12456#
EHT344  070634      3129   12456#
EHT353  070646      3178   12456#
EH1     077257     12456#
EH110   077276     12456#
EH111   077305     12456#
EH114   077324     12456#
EH223   077353     12456#
EH256   077401     12456#
EH336   077456     12456#
EH337   077515     12456#
EH344   077654     12456#
EH353   100014     12456#
EMS1    071030     12456#
EMS10   071364     12456#
EMS100  073604     12456#
EMS101  073632     12456#
EMS102  073660     12456#
```

```
EMS103  073707          12456#
EMS104  073716          12456#
EMS105  073745          12456#
EMS106  073774          12456#
EMS11   071426          12456#
EMS110  074013          12456#
EMS111  074042          12456#
EMS112  074053          12456#
EMS113  074137          12456#
EMS114  074167          12456#
EMS115  074213          12456#
EMS116  074220          12456#
EMS117  074250          12456#
EMS12   071437          12456#
EMS120  074272          12456#
EMS121  074307          12456#
EMS122  074352          12456#
EMS123  074407          12456#
EMS124  074452          12456#
EMS125  074504          12456#
EMS126  074547          12456#
EMS127  074612          12456#
EMS13   071500          12456#
EMS130  074651          12456#
EMS131  074707          12456#
EMS132  074747          12456#
EMS133  074773          12456#
EMS134  075016          12456#
EMS135  075060          12456#
EMS136  075122          12456#
EMS137  075147          12456#
EMS14   071511          12456#
EMS140  075204          12456#
EMS141  075225          12456#
EMS142  075253          12456#
EMS143  075270          12456#
EMS144  075316          12456#
EMS145  075336          12456#
EMS146  075355          12456#
EMS147  075374          12456#
EMS15   071522          12456#
EMS150  075411          12456#
EMS151  075430          12456#
EMS152  075454          12456#
EMS153  075500          12456#
EMS154  075522          12456#
EMS155  075536          12456#
EMS156  075556          12456#
EMS157  075574          12456#
EMS16   071544          12456#
EMS160  075611          12456#
EMS161  075626          12456#
EMS162  075656          12456#
EMS163  075711          12456#
EMS164  075735          12456#
EMS165  076016          12456#
```

```
EMS166  076032          12456#
EMS167  076057          12456#
EMS17   071574          12456#
EMS170  076104          12456#
EMS171  076133          12456#
EMS172  076166          12456#
EMS173  076206          12456#
EMS174  076223          12456#
EMS175  076231          12456#
EMS176  076257          12456#
EMS177  076306          12456#
EMS2    071077          12456#
EMS20   071634          12456#
EMS200  076330          12456#
EMS201  076345          12456#
EMS202  076413          12456#
EMS203  076443          12456#
EMS204  076456          12456#
EMS205  076504          12456#
EMS206  076516          12456#
EMS207  076543          12456#
EMS21   071657          12456#
EMS210  076570          12456#
EMS211  076601          12456#
EMS212  076617          12456#
EMS213  076642          12456#
EMS214  076665          12456#
EMS215  076703          12456#
EMS216  076724          12456#
EMS217  076754          12456#
EMS22   071702          12456#
EMS220  077007          12456#
EMS221  077027          12456#
EMS222  077077          12456#
EMS223  077130          12456#
EMS224  077205          12456#
EMS23   071716          12456#
EMS24   071744          12456#
EMS25   071773          12456#
EMS26   072010          12456#
EMS27   072031          12456#
EMS3    071114          12456#
EMS30   072042          12456#
EMS31   072052          12456#
EMS32   072101          12456#
EMS33   072130          12456#
EMS34   072156          12456#
EMS35   072227          12456#
EMS36   072256          12456#
EMS37   072305          12456#
EMS4    071157          12456#
EMS40   072333          12456#
EMS41   072362          12456#
EMS42   072410          12456#
EMS43   072437          12456#
EMS44   072466          12456#
```

```
EMS45   072541          12456#
EMS46   072604          12456#
EMS47   072633          12456#
EMS5    071222          12456#
EMS50   072662          12456#
EMS51   072711          12456#
EMS52   072737          12456#
EMS53   072751          12456#
EMS54   072763          12456#
EMS55   073005          12456#
EMS56   073034          12456#
EMS57   073111          12456#
EMS6    071252          12456#
EMS60   073137          12456#
EMS61   073152          12456#
EMS62   073201          12456#
EMS63   073217          12456#
EMS64   073245          12456#
EMS65   073263          12456#
EMS66   073331          12456#
EMS67   073401          12456#
EMS7    071317          12456#
EMS70   073426          12456#
EMS71   073440          12456#
EMS72   073453          12456#
EMS73   073463          12456#
EMS74   073500          12456#
EMS75   073517          12456#
EMS76   073525          12456#
EMS77   073556          12456#
EMTVEC= 000030            945#   3245*   3246*
EMT1    064434           1524   12456#
EMT10   064504           1573   12456#
EMT100  065502           1970   12456#
EMT101  065520           1977   12456#
EMT102  065536           1984   12456#
EMT103  065546           1991   12456#
EMT104  065560           1998   12456#
EMT105  065576           2005   12456#
EMT106  065606           2012   12456#
EMT107  065616           2019   12456#
EMT11   064510           1580   12456#
EMT110  065636           2026   12456#
EMT111  065650           2033   12456#
EMT112  065656           2040   12456#
EMT113  065664           2047   12456#
EMT114  065700           2054   12456#
EMT115  065706           2061   12456#
EMT116  065716           2068   12456#
EMT117  065726           2075   12456#
EMT12   064514           1587   12456#
EMT120  065736           2082   12456#
EMT121  065746           2089   12456#
EMT122  065756           2096   12456#
EMT123  065766           2103   12456#
EMT124  065776           2110   12456#
```

```
EMT125  066006            2117   12456#
EMT126  066016            2124   12456#
EMT127  066030            2131   12456#
EMT13   064522            1594   12456#
EMT130  066042            2138   12456#
EMT131  066054            2145   12456#
EMT132  066066            2152   12456#
EMT133  066100            2159   12456#
EMT134  066112            2166   12456#
EMT135  066124            2173   12456#
EMT136  066136            2180   12456#
EMT137  066150            2187   12456#
EMT14   064534            1601   12456#
EMT140  066160            2194   12456#
EMT141  066170            2201   12456#
EMT142  066200            2208   12456#
EMT143  066210            2215   12456#
EMT144  066220            2222   12456#
EMT145  066230            2229   12456#
EMT146  066240            2236   12456#
EMT147  066250            2243   12456#
EMT15   064542            1608   12456#
EMT150  066260            2250   12456#
EMT151  066270            2257   12456#
EMT152  066302            2264   12456#
EMT153  066314            2271   12456#
EMT154  066332            2278   12456#
EMT155  066350            2285   12456#
EMT156  066362            2292   12456#
EMT157  066374            2299   12456#
EMT16   064550            1615   12456#
EMT160  066406            2306   12456#
EMT161  066416            2313   12456#
EMT162  066430            2320   12456#
EMT163  066442           12456#
EMT164  066452            2335   12456#
EMT165  066460            2342   12456#
EMT166  066466            2349   12456#
EMT167  066474            2356   12456#
EMT17   064560            1622   12456#
EMT170  066502           12456#
EMT171  066512            2370   12456#
EMT172  066522            2377   12456#
EMT173  066532            2384   12456#
EMT174  066542            2391   12456#
EMT175  066554            2398   12456#
EMT176  066562            2405   12456#
EMT177  066570            2412   12456#
EMT2    064440            1531   12456#
EMT20   064570            1629   12456#
EMT200  066602            2419   12456#
EMT201  066614            2426   12456#
EMT202  066626            2433   12456#
EMT203  066640            2440   12456#
EMT204  066652            2447   12456#
EMT205  066670            2454   12456#
```

```
EMT206  066700            2461   12456#
EMT207  066710            2468   12456#
EMT21   064600            1636   12456#
EMT210  066722            2475   12456#
EMT211  066730            2482   12456#
EMT212  066744            2489   12456#
EMT213  066760            2496   12456#
EMT214  066770            2503   12456#
EMT215  067010            2510   12456#
EMT216  067022            2517   12456#
EMT217  067032            2524   12456#
EMT22   064610            1643   12456#
EMT220  067042            2531   12456#
EMT221  067052            2538   12456#
EMT222  067062            2545   12456#
EMT223  067072            2552   12456#
EMT224  067102           12456#
EMT225  067104           12456#
EMT226  067106           12456#
EMT227  067110           12456#
EMT23   064620            1650   12456#
EMT230  067112           12456#
EMT231  067114           12456#
EMT232  067116           12456#
EMT233  067120           12456#
EMT234  067122           12456#
EMT235  067124           12456#
EMT236  067126           12456#
EMT237  067130           12456#
EMT24   064630            1657   12456#
EMT240  067132           12456#
EMT241  067134           12456#
EMT242  067136           12456#
EMT243  067140           12456#
EMT244  067142           12456#
EMT245  067144           12456#
EMT246  067146            2685   12456#
EMT247  067156            2692   12456#
EMT25   064640            1664   12456#
EMT250  067170            2699   12456#
EMT251  067204            2706   12456#
EMT252  067212            2713   12456#
EMT253  067220            2720   12456#
EMT254  067236            2727   12456#
EMT255  067246            2734   12456#
EMT256  067256            2741   12456#
EMT257  067266            2748   12456#
EMT26   064650            1671   12456#
EMT260  067300            2755   12456#
EMT261  067312            2762   12456#
EMT262  067332            2769   12456#
EMT263  067342            2776   12456#
EMT264  067352            2783   12456#
EMT265  067372            2790   12456#
EMT266  067412            2797   12456#
EMT267  067424            2805   12456#
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
EMT27   064660            1678   12456#
EMT270  067442            2812   12456#
EMT271  067456            2820   12456#
EMT272  067474            2827   12456#
EMT273  067512            2834   12456#
EMT274  067530            2842   12456#
EMT275  067546            2850   12456#
EMT276  067560            2858   12456#
EMT277  067574            2867   12456#
EMT3    064446            1538   12456#
EMT30   064670            1685   12456#
EMT300  067612            2875   12456#
EMT301  067632            2883   12456#
EMT302  067654            2890   12456#
EMT303  067666            2897   12456#
EMT304  067700            2904   12456#
EMT305  067712            2911   12456#
EMT306  067724            2918   12456#
EMT307  067734            2925   12456#
EMT31   064700            1692   12456#
EMT310  067754            2932   12456#
EMT311  067766            2939   12456#
EMT312  070000            2946   12456#
EMT313  070012            2953   12456#
EMT314  070032            2960   12456#
EMT315  070044            2967   12456#
EMT316  070056            2974   12456#
EMT317  070070            2981   12456#
EMT32   064710            1699   12456#
EMT320  070100            2988   12456#
EMT321  070110            2 35   12456#
EMT322  070120            3002   12456#
EMT323  070126            3009   12456#
EMT324  070136            3016   12456#
EMT325  070150            3023   12456#
EMT326  070162            3030   12456#
EMT327  070200            3037   12456#
EMT33   064720            1706   12456#
EMT330  070212            3044   12456#
EMT331  070222            3051   12456#
EMT332  070234            3058   12456#
EMT333  070246            3065   12456#
EMT334  070264            3072   12456#
EMT335  070302            3079   12456#
EMT336  070322            2328    3086   12456#
EMT337  070332            3093   12456#
EMT34   064730            1713   12456#
EMT340  070342            3100   12456#
EMT341  070354            3107   12456#
EMT342  070362            3114   12456#
EMT343  070374            3121   12456#
EMT344  070406            3128   12456#
EMT345  070420            3135   12456#
EMT346  070430            3142   12456#
EMT347  070444            3149   12456#
EMT35   064740            1720   12456#
```

```
EMT350  070456            3156   12456#
EMT351  070474            3163   12456#
EMT352  070504            3170   12456#
EMT353  070510            3177   12456#
EMT354  070520            3184   12456#
EMT36   064750            1727   12456#
EMT37   064760            1734   12456#
EMT4    064454            1545   12456#
EMT40   064770            1741   12456#
EMT41   064776            1748   12456#
EMT42   065006            1755   12456#
EMT43   065020            1762   12456#
EMT44   065030            1769   12456#
EMT45   065040            1776   12456#
EMT46   065050            1783   12456#
EMT47   065060            1790   12456#
EMT5    064462            1552   12456#
EMT50   065066            1797   12456#
EMT51   065076            1804   12456#
EMT52   065110            1811   12456#
EMT53   065126            1819   12456#
EMT54   065144            1826   12456#
EMT55   065154            1833   12456#
EMT56   065166            1840   12456#
EMT57   065204            1847   12456#
EMT6    064470            1559   12456#
EMT60   065214            1855   12456#
EMT61   065232            1863   12456#
EMT62   065252            1871   12456#
EMT63   065266           12456#
EMT64   065270            1885   12456#
EMT65   065310            1892   12456#
EMT66   065330            1899   12456#
EMT67   065342            19??   12456#
EMT7    064476            15bb   12456#
EMT70   065354            1913   12456#
EMT71   065364            1920   12456#
EMT72   065374            1927   12456#
EMT73   065412            1935   12456#
EMT74   065430            1942   12456#
EMT75   065440            1949   12456#
EMT76   065460            1956   12456#
EMT77   065470            1963   12456#
ENRGDT  064434           12456#
EQUALS  063045           12456#
ERR     = 040000          1020#  9175   9177   9193   9227
ERRNMB  033230            7734*  7735*  7738   7750   7755   7847#
ERRTYP  032472            7713#  11985
ERRVEC= 000004             938#  3259   3260*  3271*  3506   3507   3508*  3509*  3520*  3521*  3525*  3526*  3552
                          3553   3554*  3555*  3559*  3560*  3571*  3572*  8747   8748   8752*  8753*  8784*  8785*
                          8820   8821   8825*  8826*  8851*  8852*  8863   8864   8865*  8866*  8872*  8880*  8881*
                          8895   8896   8897*  8898*  8922*  8923*  9770   9771   9774*  9775*  9808*  9809*  10056
                         10057  10058*  10059*  10070*  10071*  11874  11875*  11877*  11880*
ERTY00  033236            7720   7852#
ERTY01  033244            7728   7853#
ERTY02  033254            7737   7855#
```

```
ERTY03  033263          7743    7857#
ERTY04  033271          7839    7859#
ESRC   = 004000         1094#
FER    = 000020         1044#   1050    7111    7121    7249    7259    7400    7410    8272    9507    11105   11122   11125
                        11148   11151
FIND   = 000001         3638#   3640#   3675#   3676    3707#   3708    3714#   3715    3721#   3722    3750#   3751    3757#
                        3758    3764#   3765    3773#   3776    3806#   3808#   3843#   3844    3875#   3876    3882#   3883
                        3889#   3890    3918#   3919    3925#   3926    3932#   3933    3941#   3944    3974#   3976#   4011#
                        4012    4044#   4045    4051#   4052    4058#   4059    4088#   4089    4095#   4096    4102#   4103
                        4111#   4114    4144#   4146#   4181#   4182    4213#   4214    4220#   4221    4227#   4228    4255#
                        4256    4262#   4263    4269#   4270    4304#   4306#   4341#   4342    4373#   4374    4380#   4381
                        4387#   4388    4418#   4419    4429#   4430    4492#   4494#   4529#   4530    4561#   4562    4568#
                        4569    4575#   4576    4604#   4605    4611#   4612    4618#   4619    4627#   4630    4660#   4662#
                        4697#   4698    4729#   4730    4736#   4737    4743#   4744    4771#   4772    4778#   4779    4785#
                        4786    4820#   4822#   4857#   4858    4889#   4890    4896#   4897    4903#   4904    4934#   4935
                        4945#   4946    5008#   5010#   5045#   5046    5077#   5078    5084#   5085    5091#   5092    5119#
                        5120    5126#   5127    5133#   5134    5167#   5169#   5204#   5205    5236#   5237    5243#   5244
                        5250#   5251    5278#   5279    5285#   5286    5292#   5293    5326#   5328#   5363#   5364    5395#
                        5396    5402#   5403    5409#   5410    5437#   5438    5444#   5445    5451#   5452    5485#   5487#
                        5524#   5525    5557#   5558    5564#   5565    5571#   5572    5599#   5600    5606#   5607    5613#
                        5614    5650#   5652#   5688#   5689    5695#   569?    5702#   5703    5731#   5732    5738#   5739
                        5745#   5746    5754#   5757    5795#   5797#   5833#   5834    5840#   5841    5847#   5848    5876#
                        5877    5883#   5884    5890#   5891    5899#   5902    5940#   5942#   5978#   5979    5985#   5986
                        5992#   5993    6021#   6022    6028#   6029    6035#   6036    6044#   6047    6073#   6075#   6121#
                        6122    6128#   6129    6135#   6136    6157#   6159#   6203#   6204    6210#   6211    6217#   6218
                        6252#   6254#   6290#   6291    6297#   6298    6304#   6305    6313#   6315#   6344#   6345    6351#
                        6352    6358#   6359    6399#   6401#   6437#   6438    6444#   6445    6451#   6452    6460#   6462#
                        6491#   6492    6498#   6499    6505#   6506    6546#   6548#   6584#   6585    6591#   6592    6598#
                        6599    6607#   6609#   6638#   6639    6645#   6646    6652#   6653    6692#   6694#   6742#   6743
                        6749#   6750    6756#   6757    6804#   6805    6811#   6812    6818#   6819    6827#   6830    6867#
                        6869#   6918#   6919    6932#   6933    6941#   6943#   6972#   6973    6986#   6987    7020#   7022#
                        7058#   7059    7065#   7066    7072#   7073    7102#   7103    7113#   7114    7124#   7125    7158#
                        7160#   7196#   7197    7203#   7204    7210#   7211    7240#   7241    7251#   7252    7262#   7263
                        7297#   7299#   7344#   7345    7351#   7352    7358#   7359    7387#   7388    7402#   7403    7417#
                        7418    7437#   7438    7455#   7456    7501#   7503#   7548#   7549    7555#   7556    7562#   7563
                        7591#   7592    7602#   7603    7620#   7621
FMT16  = 010000         1119#   3796    3964    4134    4294    4482    4650    4810    4998    5157    5316    5475    5640
                        5785    5930    6084    6168    6389    6536    6682    6857    7082    7148    7287    7396    7491
                        8108    8537    8539    8610    9867    10943
FNCDTB  063676          9221    12456#
FNCMSK= 000077           962#   10654   11098
F0     = 000002          960#   9120
F1     = 000004          959#   9120
F2     = 000010          958#   9120    9694
F3     = 000020          957#   9120
F4     = 000040          956#   9120
GEMBUF  036114          3634    3802    3970    4140    4300    4488    4656    4816    5004    5163    5322    5481    5646
                        5791    5936    6248    6395    6542    6688    6863    7016    7154    7293    7497    8521#
GET     037012          3599    3667    3699    3742    3835    3867    3910    4003    4036    4080    4173    4205    4247
                        4333    4365    4410    4450    4521    4553    4596    4689    4721    4763    4849    4881    4926
                        4965    5037    5069    5111    5196    5228    5270    5355    5387    5429    5516    5549    5591
                        5680    5723    5825    5868    5970    6013    6113    6195    6282    6336    6429    6483    6576
                        6630    6734    6796    6910    6964    7050    7094    7188    7232    7336    7379    7540    7583
                        7931    7955    7982    8005    8031    8135    8168    8204    8238    8738#
GETBUF  001326          1426#   8750
GETINX  001504          1497#   3596*   3597*   4448*   4449*   4963*   4964*   7953*   7954*   8003*   8004*   8696    8751
```

```
GETSTS  036726          3662   3830   3998   4168   4328   4516   4684   4844   5032   5191   5350   5511   5676
                        5821   5966   6110   6191   6278   6332   6425   6479   6572   6626   6730   6906   6960
                        7046   7184   7332   7536   7930   7981   8024   8122   8692#
GNS     = ****** U      1269   3297   7672   7679  12340  12341  12342  12343  12344  12345  12347  12349  12350
                       12351  12352  12353  12354
GO      = 000001         961#  3647   3684   3731   3815   3852   3899   3983   4021   4069   4153   4190   4237
                        4313   4350   4400   4501   4538   4585   4669   4706   4753   4829   4866   4916   5017
                        5054   5101   5176   5213   5260   5335   5372   5419   5496   5534   5581   5659   5712
                        5804   5857   5949   6002   6087   6171   6261   6322   6408   6469   6555   6616   6685
                        6701   6716   6764   6780   6887   6950   7029   7083   7167   7221   7315   7368   7519
                        7572   7965   8014   8126   8159   8193   8227   8908   9019   9031   9036   9052   9142
                        9145  11099
GTSWR = 104407          3292   3481  12347#
HCE     = 000200        1041#  1050   7435   7445   7600   7610   8272   947u   9475   9490   9492   9505   9520
                       11105  11135  11138  11148  11151  12456
HCI     = 002000        1121#  9468  11101
HCRC    = 000400        1040#  1050   8272   9477  11105  11109  11112  11148  11151
HELP    = 102574        3345  12456#
HELPQS  = 063055        3339  12456#
HT      = 000011         848#  7776  11811  11852
IAE     = 002000        1038#  9374   9671   9686   9712   9858   9875  10374  10428  10431  10932  10947  12456
IDXMSK= 000077          1197#  8764   8767   8837   8840
IE      = 000100        1211#  1244#
ILF     = 000001        1048#  9246   9248  10268  10328  10332  10374  10395  10398  10888  10905  10908  12456
ILF02 = 000002           967#
ILF24 = 000024           977#
ILF26 = 000026           978#
ILF30 = 000030           980#
ILF32 = 000032           981#
ILF34 = 000034           982#
ILF36 = 000036           983#
ILF40 = 000040           984#
ILF42 = 000042           985#
ILF44 = 000044           986#
ILF46 = 000046           987#
ILF54 = 000054           990#
ILF56 = 000056           991#
ILF64 = 000064           994#
ILF66 = 000066           995#
ILF74 = 000074           998#
ILF76 = 000076           999#
ILR     = 000002        1047#  10268  10314  10318  10582  10586  10589  10888  10891  10894
ILRG50= 000050          1183#
ILRG52= 000052          1184#
ILRG54= 000054          1185#
ILRG56= 000056          1186#
ILRG60= 000060          1187#
ILRG62= 000062          1188#
ILRG64= 000064          1189#
ILRG66= 000066          1190#
ILRG70= 000070          1191#
ILRG72= 000072          1192#
ILRG74= 000074          1193#
ILRG76= 000076          1194#
IOTVEC= 000020           943#  3243*  3244*
IPCKO = 000001          1248#
```

```
IPCK1 = 000002          1247#
IPCK2 = 000004          1246#
IPCK3 = 000010          1245#
IR    = 000100          1225#  10131
IVC   = 010000          1154#   6925   6929   6979   6983   9289   9290   9893  10016  10442  10449  10452  10521
                       10871  10875  11303  11384  12456
LBC   = 002000          1156#  10989  11007  11012
LBT   = 002000          1024#   9449   9673   9675
LF    = 000012           849#   7774  11555  11846  11852  12456
LS    = 000004          1103#  10155  10707
LSC   = 004000          1155#  10989  11023  11026
LST   = 000002          1104#  10155  10707
MCLK  = 004000          1076#
MCPE  = 020000          1205#   9019   9036   9049   9052  10786  10789
MDF   = 000100          1081#
MDPE  = 000400          1222#   9545  11195  11198
MEDENB  001472          1485#  3492*  7964*  8087   8343*
MFGFIL  100554          8109   8312   8325*  8332*  8393  12456#
MI    = 000004          1084#
MIXED   064006         12456#
MOC   = 000400          1079#
MOH   = 020000          1111#
MOL   = 010000          1022#   8264   9954   9964   9965   9970   9975  10003  10254  10257  10418  10495  10496
                       10503  10508  10667  10848  11009  11262  11263  11285  11290  11361  11362  11366  11371
MRD   = 002000          1077#
MS    = 000040          1082#
MSC   = 000002          1085#
MSE   = 100000          1162#   7394   8312   8535
MSER  = 000200          1080#
MUR   = 001000          1078#
MWD   = 000010          1102#  10156  10708
MWP   = 000010          1083#
MXF   = 001000          1221#   9434  10809  10823  10826
NOTMSK= 115760          1050#
NED   = 010000          1218#   3561   9001   9005   9783
NEM   = 004000          1219#   9419
NOP   = 000000           966#
NSA   = 100000          1109#
OCC   = 100000          1090#
OFD   = 000200          1122#   6835   6837
OFFSET= 000014           972#   6701   6764
OM    = 000001          1029#  10495  10551  10554  10666  11210  11213
ONES    064046         4486    4654   4814  12456#
OPE   = 020000          1153#
OPI   = 020000          1035#   9973   9948   9972  10268  10286  10290  10374  10411  10414  10505  10843  10846
                       11287  11368  12456
OR    = 000200          1224#
PACACK= 000022           976#   8159
PAKACK= 000022           975#    976   7965
PAR   = 000010          1045#   8275   9064   9068   9080   9840  10374  10379  10384  10800  10805
PAT   = 000020          1227#
PDA   = 000400          1097#
PGE   = 002000          1220#
PGM   = 001000          1025#  10666
PHA   = 000200          1098#
PIP   = 020000          1021#   8012   8190   9964   9985   9990  10495  10566  10571  10666  11262  11268  11273
```

```
                    11361  11398  11403
PIRQ   = 177772      855#
PIRQVE= 000240       949#
PLFS   = 002000     1095#
PRIERR  040006      3707   3750   3875   3918   4044   4088   4213   4255   4373   4418   4561   4604   4729
                    4771   4889   4934   5077   5119   5236   5278   5395   5437   5557   5599   5688   5731
                    5833   5876   5978   6021   6121   6203   6290   6344   6437   6491   6584   6638   6742
                    6804   6918   6972   7058   7102   7196   7240   7344   7387   7548   7591   8967#
PROMPT  063047      3418   3426  12456#
PR0    = 000000      872#
PR1    = 000040      873#
PR2    = 000100      874#
PR3    = 000140      875#  11534
PR4    = 000200      876#
PR5    = 000240      877#
PR6    = 000300      878#  3230   3487   3509   3555   8753   8826   8866   8898   9775  10059  11540
PR7    = 000340      879#
PS     = 177776      852#   853
PSEL   = 002000     1207#
PSW    = 177776      853#
PUT     037262      3654   3690   3733   3822   3858   3901   3990   4027   4071   4160   4196   4238   4320
                    4356   4401   4508   4544   4587   4676   4712   4754   4836   4872   4917   5024   5060
                    5102   5183   5219   5261   5342   5378   5420   5503   5540   5582   5668   5714   5813
                    5859   5958   6004   6102   6183   6270   6324   6417   6471   6564   6618   6708   6722
                    6771   6787   6879   6898   6952   7038   7085   7176   7223   7324   7370   7528   7574
                    7968   8017   8127   8160   8194   8228   8814#
PUTBUF  001376      1453#  8098   8347*  8823
PUTINX  001533      1502#  3648   3685   3816   3853   3984   4022   4154   4191   4314   4351   4502   4539
                    4670   4707   4830   4867   5018   5055   5177   5214   5336   5373   5497   5535   5660
                    5805   5950   6089   6172   6262   6409   6556   6702   6717   6765   6782   6877*  6878*
                    6889   7030   7168   7316   7520   7966*  7967*  8015*  8016*  8111   8824   9071
PWRVEC= 000024       944#  3249*  3250*  12359*  12360*  12369*  12375*  12387*  12388*
QSTMRK  063053      3334   3436  12456#
RCLSTS  047746      8038   8213  10365#
RD     = 000070      996#
RDCHR  = 104411     3326   3340   3350   3419   3427  12212  12350#
RDLIN  = 104412    12282  12351#
RDOCT  = 104413     3362   3378   3399  12352#
RDY    = 000200     1210#  8908   8909   9015   9018  10106
READY   006764      3331   3485#  7647   7695
RECAL  = 000006      969#  8014   8193
RESREG= 104415      7843  12354#
RESVEC= 000010       939#
REX    = 010000     1093#
RG     = 040000     1091#
RGOTPT  064006     12456#
RH     = 000072      997#  3731   3899   4069   4585   5712   5857   6002   6087   6171   6322   6469   6616
                    6780   6950   7083   7221   7368   7572   8227
RIP    = 000020      974#
RLEASE= 000012       971#
RMAS   = 000016     1171#
RMASI   001344      1436#  10687  12456
RMASO   001414      1463#
RMBA   = 000004     1253#  3514*  3515   3687   3855   4024   4193   4353   4541   4709   4869   5057   5216
                    5375   5537   5665   5810   5955   6093   6177   6267   6414   6561   6719   6784   6895
                    7035   7173   7321   7525   8116
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
RMBAE = 000050          1256#
RMBAI   001332          1431#  4444   4959   9621   9623  10117  12456
RMBAO   001402          1458#  3630*  3732*  3798*  3900*  3966*  4070*  4136*  4296*  4484*  4586*  4652*  4812*
                        5000*  5159*  5318*  5477*  5642*  5713*  5787*  5858*  5932*  6003*  6086*  6170*  6244*
                        6323*  6391*  6470*  6538*  6617*  6684*  6781*  6838*  6859*  6951*  7012*  7084*  7150*
                        7222*  7289*  7369*  7493*  7573*  8109*  8527   8614   9617   9620
RMCC  = 000036          1178#
RMCCI   001364          1444#
RMCCO   001434          1471#
RMCS1 = 000000          1167#  1251#  3511*  3557   3652   3688   3820   3856   3988   4025   4158   4194   4318
                        4354   4506   4542   4674   4710   4834   4870   5022   5058   5181   5217   5340   5376
                        5501   5538   5666   5811   5956   6095   6178   6268   6415   6562   6706   6720   6769
                        6785   6896   7036   7174   7322   7526   7966   8015   8117   8755   8828   8907   9781
RMCS1I  001326          1429#  8994   8996   8998   9015   9017   9020   9031   9035   9037   9049   9051   9053
                        9142   9144   9155   9198   9604  10104  10653  10786  10788  10790  10809  12456
RMCS10  001376          1456#  3631*  3647*  3684*  3731*  3799*  3815*  3852*  3899*  3967*  3983*  4021*  4069*
                        4137*  4153*  4190*  4237*  4297*  4313*  4350*  4400*  4485*  4501*  4538*  4585*  4653*
                        4669*  4706*  4753*  4813*  4829*  4866*  4916*  5001*  5017*  5054*  5101*  5160*  5176*
                        5213*  5260*  5319*  5335*  5372*  5419*  5478*  5496*  5534*  5581*  5643*  5659*  5712*
                        5788*  5804*  5857*  5933*  5949*  6002*  6087*  6171*  6245*  6261*  6322*  6392*  6408*
                        6469*  6539*  6555*  6616*  6685*  6701*  6716*  6764*  6780*  6860*  6887*  6950*  7013*
                        7029*  7083*  7151*  7167*  7221*  7290*  7315*  7368*  7494*  7519*  7572*  7965*  8014*
                        8126*  8159*  8193*  8227*  8373   8531   9119   9635  11077  11097
RMCS2 = 000010          1254#  3516*  3517   3558   8754   8827   9780*  9782  10061*  10063*
RMCS2I  001336          1433#  4427   4435   4437   4943   4951   4953   8976   9001   9003   9004   9190   3261
                        9362   9405   9420   9435   9546   9618  10126  10823  10825  10827  11195  11197  11199
                       11249  11251  11253  12456
RMCS20  001406          1460#
RMCS3 = 000052          1257#
RMDA  = 000006          1168#  3649   3817   3985   4155   4315   4503   4671   4831   5019   5178   5337   5498
                        5661   5806   5951   6090   6173   6263   6410   6557   6703   6766   6891   7031   7169
                        7317   7521   8112
RMDAI   001334          1432#  9717  12456
RMDAO   001404          1459#  3627*  3795*  3963*  4133*  4293*  4481*  4649*  4809*  4997*  5156*  5315*  5474*
                        5639*  5644   5764*  5765   5784*  5789   5909*  5910   5929*  6083*  6167*  6241*  6246
                        6368*  6369   6388*  6393   6515*  6516   6535*  6681*  6686   6856*  6861   7009*  7147*
                        7286*  7490*  8105*  8286*  8287   8289   8307   8315*  8367   8494*  8530   9643   9644
                        9863   9865   9869  10937  10939  10941
RMDB  = 000022          1255#  3518*  3519   4448   4963   8703
RMDBI   001350          1438#  4455   4970
RMDBO   001420          1465#
RMDC  = 000034          1177#  3650   3818   3986   4156   4316   4504   4672   4832   5020   5179   5338   5499
                        5662   5807   5952   6091   6174   6264   6411   6558   6704   6767   6892   7032   7170
                        7318   7522   8113
RMDCI   001362          1443#  9729  12456
RMDCO   001432          1470#  3626*  3794*  3962*  4132*  4292*  4480*  4648*  4808*  4996*  5155*  5314*  5473*
                        5494   5495*  5533*  5638*  5783*  5928*  5934   6054*  6055   6082*  6166*  6240*  6387*
                        6534*  6540   6661*  6662   6680*  6855*  7008*  7146*  7285*  7489*  8106*  8366   8493*
                        8529   9645   9860  10934
RMDS  = 000012          1169#  7953   8003
RMDSI   001340          1434#  7962   8012   8157   8190   8264   9033   9129   9131   9176   9193   9227   9231
                        9287   9314   9449   9674   9899   9954   9963   9970   9974   9976   9985   9989   9991
                       10000  10002  10004  10014  10018  10020  10199  10240  10242  10244  10254  10256  10258
                       10418  10456  10494  10503  10507  10509  10519  10523  10524  10535  10537  10539  10551
                       10553  10555  10566  10570  10572  10665  10848  10850  10879  11009  11075  11210  11212
                       11214  11261  11268  11272  11274  11285  11289  11291  11301  11305  11307  11360  11366
```

```
                    11370  11372  11382  11386  11387* 11388  11398  11402  11404  12456
RMDSO   001410       1461#
RMDT  = 000026       1174#  3596
RMDTI   001354       1440#  3598*  3603   3605   12456
RMDTO   001424       1467#
RMEC1 = 000044       1181#
RMEC1I  001372       1447#  8615   12456
RMEC1O  001442       1474#
RMEC2 = 000046       1182#  8698
RMEC2I  001374       1448#  8636   8697   10167  10731  12456
RMEC2O  001444       1475#
RMER1 = 000014       1170#  9072
RMER1I  001342       1435#  7111   7119   7120   7249   7257   7258   7400   7408   7409   7435   7443   7444
                     7600   7608   7609   8271   8604   8606   9064   9079   9081   9171   9247   9274   9317
                     9332   9375   9452   9478   9493   9508   9561   9580   9583   9671   9686   9701   9712
                     9840   9847   9874   9948   9951   9972   10144  10268  10272  10274  10275  10286  10288
                    10289  10300  10302  10303  10314  10316  10317  10328  10330  10331  10374  10379  10383
                    10385  10395  10397  10399  10411  10413  10415  10428  10430  10432  10505  10582  10586
                    10588  10590  10600  10602  10604  10614  10618  10620  10678  10800  10804  10806  10843
                    10845  10847  10888  10891  10893  10895  10905  10907  10909  10919  10921  10923  10946
                    11036  11039  11043  11045  11054  11056  11058  11068  11070  11071  11105  11109  11111
                    11113  11122  11124  11126  11135  11137  11139  11148  11150  11152  11170  11172  11174
                    11180  11236  11238  11240  11287  11368  11479  12456
RMER1O  001412       1462#  9068
RMER2 = 000042       1180#
RMER2I  001370       1446#  6925   6927   6928   6979   6981   6982   7415   7424   7425   8273   8278   9066
                     9173   9291   9347   9523   9842   9893   9896   9907   9935   9938   9987   10016  10189
                    10381  10442  10449  10451  10453  10466  10468  10470  10480  10482  10484  10521  10568
                    10616  10740  10802  10871  10874  10876  10960  10964  10989  10993  10995  10997  11007
                    11011  11013  11023  11025  11027  11041  11223  11225  11227  11270  11303  11384  11400
                    11414  11420  11422  11424  11434  11436  11438  11482  12456
RMER2O  001440       1473#
RMLA  = 000020       1172#
RMLAI   001346       1437#  12456
RMLAO   001416       1464#
RMMR1 = 000024       1173#  6877   6890
RMMR1I  001352       1439#  10154  10706  12456
RMMR1O  001422       1466#  6876*  6888*
RMMR2 = 000040       1179#
RMMR2I  001366       1445#  10176  10718  12456
RMMR2O  001436       1472#
RMOF  = 000032       1176#  3651   3819   3987   4157   4317   4505   4673   4833   5021   5180   5339   5500
                     5663   5808   5953   6092   6175   6265   6412   6559   6705   6768   6893   7033   7171
                     7319   7523   8115
RMOFI   001360       1442#  8608   8610   9468   9578   11101  11177  12456
RMOFO   001430       1469#  3628*  3796*  3964*  4134*  4294*  4482*  4650*  4810*  4998*  5157*  5316*  5475*
                     5640*  5785*  5930*  6084*  6168*  6242*  6389*  6536*  6682*  6835   6837*  6857*  7010*
                     7082*  7148*  7220*  7287*  7491*  8108*  8537   9867   10943  12456
RMR   = 000004       1046#  10268  10300  10304  10582  10600  10603  10888  10919  10922
RMSN  = 000030       1175#
RMSNI   001356       1441#  12456
RMSNO   001426       1468#
RMWC  = 000002       1252#  3512*  3513   3686   3854   4023   4192   4352   4540   4708   4868   5056   5215
                     5374   5536   5664   5809   5954   6094   6176   6266   6413   6560   6718   6783   6894
                     7034   7172   7320   7524   8114
RMWCI   001330       1430#  8653   9602   9614   9632   9696   12456
```

```
RMWCO   001400          1457#  3629*  3797*  3965*  4020*  4068*  4135*  4295*  4483*  4651*  4811*  4999*  5158*
                        5317*  5476*  5641*  5786*  5931*  6085*  6169*  6243*  6390*  6537*  6683*  6858*  7011*
                        7149*  7288*  7492*  8107*  8369   8528   8654   9615   9633
RQA     = 100000        1131#  10177  10719
RQB     = 040000        1132#  10177  10719
RTC     = 000016         973#
SAOMSK= 000037          1015#
SAVREG= 104414          7714   12353#
SA1     = 000001        1010#
SA16    = 000020        1006#
SA2     = 000002        1009#
SA4     = 000004        1008#
SA8     = 000010        1007#
SC      = 100000        1203#  9019   9036   9052   10105
SCTMSG  062734         12456#
SCTMSK= 003700          1066#
SCO     = 000100        1064#
SC1     = 000200        1063#
SC2     = 000400        1062#
SC3     = 001000        1061#
SC4     = 002000        1060#
SEARCH= 000030           979#  9195   9596
SECERR  040640          3721   3764   3889   3932   4058   4102   4227   4269   4387   4575   4o18   4743   4785
                        4903   5091   5133   5250   5292   5409   5451   5571   5613   5702   5745   5847   5890
                        5992   6035   6135   6217   6304   6358   6451   6505   6598   6652   6756   6818   6932
                        6986   7072   7124   7210   7262   7358   7455   7562   7620   9115#
SEEK    = 000004         968#  3647   3815   3983   4153   4313   4501   4669   4829   5017   5176   5335   5496
SEKSTS  044714          3675   3843   4011   4181   4341   4529   4697   4857   5045   5204   5363   5524   9829#
SHUT    056660         11548#  12053  12136
SIZCLK  037500          3468   8862#
SKI     = 040000        1152#  9859   9862   9878   9908   9910   9914   9925   9987   10190  10442  10466  10469
                        10568  10933  10936  10950  10961  10963  10964  10966  10978  11270  11400  11415  11434
                        11437
SNGPRT= 020024          1114#  3603   3607
STACK   = 001100         843#  3241   3501   3541   3585   3619   3787   3955   4125   4285   4473   4641   4801
                        4989   5148   5307   5466   5631   5776   5921   6066   6151   6233   6380   6527   6673
                        6848   7001   7139   7277   7481
STANDA  006014          3305   3316#
START   005322          1289   3225#  3530   11554
STCDRV  055712         11353#
STKLMT= 177774           854#
STOP    0[6630         11531#  11868
STSD1   1[J444         12456#
STSD2   100462         12456#
STSD3   100502         12456#
STSD4   100514         12456#
STSF    100[45         12456#
STSH1   100073         12456#
STSH2   100154         12456#
STSH3   100247         12456#
STSH4   100307         12456#
SWR     001154          1342#  3239   3261*  3263   3269*  3276*  3290   3479   7715   11869  11883  11886  11901
                        11908  11976  11983  11995  11999  12056  12095  12150*  12367  12380*
SWREG   000176          1271#  3269   3290   3479   12056  12095  12118
SW0     = 000001         907#
SW00    = 000001         897#   907
```

```
SW01  = 000002            896#    906
SW02  = 000004            895#    905
SW03  = 000010            894#    904
SW04  = 000020            893#    903
SW05  = 000040            892#    902
SW06  = 000100            891#    901
SW07  = 000200            890#    900
SW08  = 000400            889#    899
SW09  = 001000            888#    898
SW1   = 000002            906#
SW10  = 002000            887#
SW11  = 004000            886#
SW12  = 010000            885#
SW13  = 020000            884#   7715
SW14  = 040000            883#
SW15  = 100000            882#
SW2   = 000004            905#
SW3   = 000010            904#
SW4   = 000020            903#
SW5   = 000040            902#
SW6   = 000100            901#
SW7   = 000200            900#
SW8   = 000400            899#
SW9   = 001000            898#
TADMSK= 003400           1014#
TAG   = 020000           1133#
TAGADR= 001114           1314#   1321
TAP   = 040000           1110#
TA1   = 000400           1005#
TA2   = 001000           1004#
TA4   = 002000           1003#
TBITVE= 000014            940#
TIMOUT  037622           3663    3698    3741    3831    3866    3909    3999    4035    4079    4169    4204    4246    4329
                         4364    4409    4517    4552    4595    4685    4720    4762    4845    4880    4925    5033    5068
                         5110    5192    5227    5269    5351    5386    5428    5512    5548    5590    5679    5722    5824
                         5867    5969    6012    6112    6194    6281    6335    6428    6482    6575    6629    6733    6795
                         6909    6963    7049    7093    7187    7231    7335    7378    7539    7582    8025    8203    8237
                         8891#
TKVEC = 000060             947#  12033*  12034*
TPVEC = 000064             948#
TRAPVE= 000034             946#   3247*   3248*
TRE   = 040000            1204#   9019    9036    9052    9197    9199    9604
TRTVEC= 000014             941#
TST   = 010000            1134#  10178   10720
TSTNMB  03322F            7726*   7727*   7729    7846#
TSTPRP  03327C            3638    3806    3974    4144    4304    4492    4660    4820    5008    5167    5326    5485    5650
                          5795    5940    6073    6157    6252    6313    6399    6460    6546    6607    6692    6867    6941
                          7020    7158    7297    7501    7894#
TSTQUE  001446            1481#   3451    3452*   3491    3503    3543    3587    3621    3789    3957    4127    4287    4475
                          4643    4803    4991    5150    5309    5468    5633    5778    5923    6068    6153    6235    6382
                          6529    6675    6850    7003    7141    7279    7483    7642    7644*   7648*   9777   10062   10129
                         10690
TST1    007016            3:.96#  11928
TST10   012532            4282#   11935
TST11   013452            4470#   11936
TST12   014252            4638#   11937
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
TST13   015026            4798#  11938
TST14   015744            4986#  11939
TST15   016520            5145#  11940
TST16   017274            5304#  11941
TST17   020050            5463#  11942
TST2    007214            3538#  11929
TST20   020652            5628#  11943
TST21   021376            5773#  11944
TST22   022122            5918#  11945
TST23   022644            6063#  11946
TST24   023142            6148#  11947
TST25   023440            6230#  11948
TST26   024166            6377#  11949
TST27   024714            6524#  11950
TST3    007400            3582#  11930
TST30   025442            6670#  11951
TST31   026334            6845#  11952
TST32   027142            6998#  11953
TST33   027664            7136#  11954
TST34   030406            7274#  11955
TST35   031414            7478#  11956
TST4    007542            3616#  11931
TST5    010342            3784#  11932
TST6    011142            3952#  11933
TST7    011756            4122#  11934
TYPBN = 104406            7836   12345#
TYPDS = 104405            7676    7683   7830   12344#
TYPE  = 104401            3284    3325   3328   3334   3339   3342   3345   3349   3352   3358   3361   3367   3372
                          3377    3383   3393   3398   3404   3417   3418   3421   3426   3429   3436   7670   7677
                          7684    7719   7720   7728   7737   7743   7752   7790   7796   7800   7813   7819   7821
                          7839    7841  11549  11621  11687  11756  11816  11978  11986  12050  12062  12116  12117
                         12120   12131  12141  12152  12171  12219  12225  12230  12234  12239  12240  12242  12245
                         12249   12340# 12389
TYPOC = 104402            3360    7833  12119  12341#
TYPON = 104404           12343#
TYPOS = 104403            3374    3795   7723   7731   7740   7746  12342#
UBUSQS  063111            3349   12456#
UNS   = 040000            1034#  10145  10268  10272  10276  10582  10614  10619  11036  11039  11044
UNTMSK= 000007            1235#   3977   8979
UPE   = 020000            1217#   9361
USE   = 040000            1163#   7394   8312   8535
USRFIL  101564            8325*   8333*  8484   8486  12456#
U0    = 000001            1232#
U1    = 000002            1231#
U2    = 000004            1230#
VV    = 000100            1028#   7962   8157   9287   9899   9964   9965  10014  10019  10240  10243  10456  10495
                         10496   10519  10525  10666  10850  10879  11262  11263  11301  11306  11361  11362  11382
                         11387
WC    = 000040            1100#  10155  10707
WCD   = 000050             988#
WCE   = 040000            1216#   4427   4436   4943   4952   9260  12456
WCEHI = 010000            1241#
WCELO = 004000            1242#
WCF   = 000040            1043#   1050   9331  11236  11239
WCH   = 000052             989#   4237   4400   4753   4916   5101   5260   5419   5581
WD    = 000060             992#
```

```
WH     = 000062            993#  3631  3684  3799  3852  3967  4021  4137  4190  4297  4350  4485  4538
                          4653  4706  4813  4866  5001  5054  5160  5213  5319  5372  5478  5534  5643
                          5659  5788  5804  5933  5949  6245  6261  6392  6408  6539  6555  6685  6716
                          6860  6887  7013  7029  7151  7167  7290  7315  7494  7519 11099
WLE    = 004000           1037#  1050  9312  9316  9329  9344  9359 11036 11068 11072 11080 12456
WRL    = 004000           1023#  9314 11075
ZEROS    064110           3632  3800  3968  4138  4298  5002  5161  5320  5479  7014  7152  7291  7495
                         12456#
$APTHD   001100           1301  1307#
$ASTAT= ****** U         12432 12447
$ATYC    062512          12403 12405#
$ATY1    062466          12401#
$ATY3    062474          11801 12402#
$ATY4    062504          11991 12404#
$AUTOB   001150           1339#  3294*  3483*  12109  12266
$BASE    001276           1408#  3359  3369*  3502  3542  3586  3620  3788  3956  4126  4286  4474  4642
                          4802  4990  5149  5308  5467  5632  5777  5922  6067  6152  6234  6381  6528
                          6674  6849  7002  7140  7278  7482  8749  8822  8899  9776 10060 12456
$BOAOR   001136           1334#  4444*  4445*  4458  4959*  4960*  4973  8664* 12456
$BODAT   001142           1336#  4437*  4455*  4462  4953*  4970*  4978  6927*  6981*  7119*  7257*  7408*  7424*
                          7443*  7608*  8662*  8976*  8977*  8980  8998*  9004*  9020*  9037*  9053*  9081*  9131*
                          9132*  9144*  9145*  9155*  9156*  9158  9176*  9177*  9178  9198*  9199*  9200  9231*
                          9232*  9233  9247*  9248*  9249  9261*  9262*  9274*  9275*  9291*  9292*  9317*  9318*
                          9332*  9333*  9347*  9348*  9362*  9363*  9375*  9376*  9405*  9406*  9420*  9421*  9435*
                          9436*  9452*  9453*  9478*  9479*  9493*  9494*  9508*  9509*  9523*  9524*  9546*  9547*
                          9561*  9562*  9583*  9584*  9602*  9623*  9674*  9675*  9676  9701*  9702*  9703  9717*
                          9718  9729*  9730  9847*  9874*  9875*  9876  9896*  9907*  9908*  9914  9925  9938*
                          9951*  9976*  9990*  9991*  10004*  10020*  10104*  10105*  10107  10117*  10126*  10134  10144*
                         10145*  10154*  10155*  10157  10167*  10176*  10177*  10179  10189*  10190*  10199*  10200*  10202
                         10244*  10258*  10274*  10288*  10302*  10316*  10330*  10385*  10399*  10415*  10432*  10453*  10470*
                         10484*  10509*  10524*  10539*  10555*  10572*  10590*  10604*  10620*  10653*  10654*  10656  10665*
                         10686*  10668  10678*  10687*  10694*  10697  10706*  10707*  10709  10718*  10719*  10721  10731*
                         10740*  10790*  10806*  10827*  10847*  10876*  10895*  10909*  10923*  10946*  10947*  10948  10960*
                         10961*  10997*  11013*  11027*  11045*  11058*  11070*  11113*  11126*  11139*  11152*  11174*  11199*
                         11214*  11227*  11240*  11253*  11274*  11291*  11307*  11372*  11388*  11404*  11424*  11438*  11494*
                         11509*  12456
$BELL    001212           1359#  11978  12011  12062  12259
$BIN     057074          11617*  11620*  11621  11628#
$CDW1    001302           1410#
$CDW2    001304           1411#
$CHARC   060026          11818*  11828*  11835  11844*  11849#
$CKSWR   061140          12095#  12349
$CMTAG   001114           1322#  3236  3237  3245  3251  3252  3253
$CM3   = 000000           1352#
$CM4   = 000005           1352#  1353#  1354#  1355#  1356#  1357#
$CNTLC   062036          12050  12131  12259#
$CNTLG   062050          12116  12261#
$CNTLU   062043          12141  12234  12260#
$CPUOP   001250           1382#
$CRLF    001217           1361#  7684  7719  7752  7790  7800  7813  7821  7841  11817  11852  11986  12011
                         12152  12239  12259
$DBLK    057312          11653  11687  11695#
$DDW0    001306           1412#
$DDW1    001310           1413#
$DDW2    001312           1414#
$DDW3    001314           1415#
```

```
$DOW4   001316          1416#
$DOW5   001320          1417#
$DOW6   001322          1418#
$DOW7   001324          1419#
$DEVCT  001232          1373#
$DEVM   001300          1409#   3311*   3416*   3424*   3442*   3443    3450
$DOAGN  032462          7666    7687    7693#
$DOTBL  057302         11656   11691#
$ENDAD  032452          1280    3282    7689#  12006
$ENDCT  032316          3251    7668#
$ENULL  032466          7696#
$ENV    001242          1378#   3288    3304    3477   11796   11988   12410   12434
$ENVM   001243          1379#   3274    3309   11798   11803   12412
$EOP    032262          3531    7658#  11552
$EOPCT  032310          3251*   7665#   7669
$EOSP   032226          3574    3610    7640#
$ERFLG  001117          1325#  11857   11897   11899   11905*  11927   11973*  12011
$ERMAX  001131          1331#   3254*  11899   11922*  11927
$ERROR  060442          3245   11971#
$ERRPC  001132          1332#   7744   11980*  11981*  11982   12011
$ERRTB  001562          1520#   7759
$ERTTL  001126          1329#   7681    7685*  11979*  12011
$ESCAP  001210          1358#   3253*  11921*  12002   12004   12011
$ETABL  001242          1377#
$ETEND  001326          1313    1420#
$FATAL  001224          1370#  12438#
$FFLG   062732         12401*  12404*  12432   12441*  12449#
$FILLC  001172          1350#  11821   11852
$FILLS  001171          1349#  11852
$GOAOR  001134          1333#   4443*   4458    4958*   4973    8663*  12456
$GOOAT  001140          1335#   4435*   4436*   4456*   4457*   4462    4951*   4952*   4971*   4972*   4978    6928*   6929*
                        6982*   6983*   7120*   7121*   7258*   7259*   7409*   7410*   7425*   7426*   7444*   7445*   7609*
                        7610*   8661*   8978*   8979*   8980    8996*   8997*   9003*   9005*   9017*   9018*   9019*   9035*
                        9036*   9051*   9052*   9079*   9080*   9133*   9146*   9157*   9158    9170*   9175*   9178    9189*
                        9197*   9200    9226*   9229*   9230*   9233    9245*   9246*   9249    9256*   9608*   9614*   9615*
                        9616*   9617*   9620*   9621    9668*   9673*   967[illegible]   9685*   9698*   9703    9714*   9715*   9716*
                        9718    9727*   9728*   9730    9846*   9858*   9871*   9876    9895*   9909*   9910*   9937*   9950*
                        9974*   9975*   9989*  10002*  10003*  10018*  10019*  10106*  10107   10116*  10132*  10134   10143*
                       10156*  10157   10166*  10178*  10179   10188*  10201*  10202   10242*  10243*  10256*  10257*  10275*
                       10276*  10289*  10290*  10303*  10304*  10317*  10318*  10331*  10332*  10383*  10384*  10397*  10398*
                       10413*  10414*  10430*  10431*  10451*  10452*  10468*  10469*  10482*  10483*  10507*  105[illegible]  10523*
                       10525*  10537*  10538*  10553*  10554*  10570*  10571*  10588*  10589*  10602*  10603*  10618*  10619*
                       10655*  10656   10667*  10668   10677*  10708*  10709   10720*  10721   10729*  10788*  10789*  10804*
                       10805*  10825*  10826*  10845*  10846*  10874*  10875*  10893*  10894*  10907*  10908*  10921*  10922*
                       10932*  10945*  10948   10962*  10963*  10995*  10996*  11011*  11012*  11025*  11026*  11043*  11044*
                       11056*  11057*  11071*  11072*  11080*  11111*  11112*  11124*  11125*  11137*  11138*  11150*  11151*
                       11172*  11173*  11197*  11198*  11212*  11213*  11225*  11226*  11238*  11239*  11251*  11252*  11272*
                       11273*  11289*  11290*  11305*  11306*  11370*  11371*  11386*  11402*  11403*  11422*  11423*  11436*
                       11437*  11495*  11510*  12456
$GET42  032442          7686#
$GTSWR  061230         12117#  12347
$HD     = 000001         819     820
$HIBTS  001100          1308#
$HIOCT  062200         12299*  12304#
$ICNT   001120          1326#  11912*  11913   11915*  11926
$ILLUP  062450         12359   12375   12392#
```

```
$INTAG  001151          1340#  12114*  12133   12153   12266
$ITEMB  001130          1330#   7735   11982*  11990   12011
$LF     001220          1362#  11852   12011   12249   12259
$LFLG   062731         12442*  12448#
$LPADR  001122          1327#   3255*   3498*  11894*  11903*  11919*  11924   11926
$LPERR  001124          1328#   3256*   3499*  11903   11920*  11926   12001
$MAOR1  001254          1395#
$MAOR2  001260          1399#
$MAOR3  001264          1402#
$MAOR4  001270          1405#
$MAIL   001222          1309    1313    1368#   3273    3288    3477    3504    3544    3588    3622    3790    3958    4128
                        4288    4476    4644    4804    4992    5151    5310    5469    5634    5779    5924    6069    6154
                        6236    6383    6530    6676    6851    7004    7142    7280    7484   11796   11918   11988
$MAMS1  001252          1389#
$MAMS2  001256          1397#
$MAMS3  001262          1400#
$MAMS4  001266          1403#
$MBADR  001102          1309#
$MFLG   062730         12402*  12408`  12443*  12447#
$MNEW   062066         12120   12264#
$MSGAD  001236          1375#  12418*  12421
$MSGLG  001240          1376#  12423*
$MSGTY  001222          1369#  12416   12424*  12436   12440*
$MSWR   062055         12117   12262#
$MTYP1  001253          1390#
$MTYP2  001257          1398#
$MTYP3  001263          1401#
$MTYP4  001267          1404#
$MXCNT  060346         11916   11926#
$NULL   001170          1348#  11823   11852
$NWTST= 000001          3493#   3495    3534#   3536    3578#   3580    3613#   3781#   3949#   4119#   4279#   4467#   4635#
                        4795#   4983#   5142#   5301#   5460#   5625#   5770#   5915#   6060#   6145#   6227#   6374#   6521#
                        6667#   6842#   6995#   7133#   7271#   7475#
$OCNT   057544         11728*  11757*  11770#
$OMODE  057546         11723*  11727*  11732   11735*  11746*  11772#
$OVER   060332         11870   11895   11904   11914   11923#
$PASS   001230          1372#   3273*   7662*   7663*   7674    7696   11910   11927
$PASTM  001106          1311#
$POWER  062456         12390   12395#
$PWRON  062310          3249   12359#  12387
$PWRMG  062444         12390#
$PWRUP  062362         12369   12375#
$QUES   001216          1360#  11852   12011   12171   12242   12259
$POCHR  061502         12184#  12350
$RODEC= ****** U       12353
$ROLIN  061572         12207#  12351
$POOCT  062100         12277#  12352
$RDSZ = 000010         12200#
$RESRE  056764         11593#  12354
$RTNAD  032464          7695#
$R2A  = ****** U       12355
$SAVRE  056726         11577#  12353
$SAVR6  062454         12368*  12376   12377*  12378*  12394#
$SCOPE  060032          3243   11866#
$SETUP= 000137          1262#   3242    3243    3245    3247    3249    3251    3252    3253    3255    3282    3285    3474
                        7660   11867   11972   11998   12006   12054   12059   12060   12090   12266
```

```
$STUP = 177777          1262#
$SVLAD  060276         11878   11917#
$SVPC = 000200          1278#   1283
$SWR  = 167400           808#    819     824     825     826     827     828     829     830     831    1357    1358    1359
                        3252    3253    3255    3256    3504    3544    3588    3622    3790    3958    4128    4288    4476
                        4644    4804    4992    5151    5310    5469    5634    5779    5924    6069    6154    6236    6383
                        6530    6676    6851    7004    7142    7280    7484    7655    7661    7688    7694    7696   11858
                       11859   11860   11861   11862   11869   11881   11883   11884   11897   11898   11899   11906   11907
                       11908   11920   11923   11926   11963   11964   11965   11966   11967   11976   11983   11995   11999
                       12011   12391
$SWREG  001244          1380#   3276
$SWRMK= 000000           831     832    11862   11863   11887
$SWOBT  060350         11893   11927#
$TESTN  001226          1371#   3504#   3544#   3588#   3622#   3790#   3958#   4128#   4288#   4476#   4644#   4804#   4992#
                        5151#   5310#   5469#   5634#   5779#   5924#   6069#   6154#   6236#   6383#   6530#   6676#   6851#
                        7004#   7142#   7280#   7484#   7727   11918#
$TIMES  001206          1357#   3252#   7661#  11906#  11913   11916#  11926
$TKB    001162          1345#  12014   12035   12046   12071   12099   12126
$TKCNT  060642         12015#  12030#  12060   12077#  12191   12193#
$TKINT  060652          3317    3486   12030#  12051   12112
$TKQEM= 060651         12019#  12085   12196
$TKQIN  060644         12016#  12031#  12032   12083#  12084#  12085   12087#
$TKQOU  060646         12017#  12032#  12194   12195#  12196   12198#
$TKQSR  060650         12018#  12031   12087   12198
$TKS    001160          1344#  12014   12036#  12067#  12069   12075#  12097   12113#  12123   12135#  12155#
$TKSRV  060722         12033   12046#
$TMP0   001174          1352#   3558#   3561    3632#   3800#   3968#   4138#   4298#   4486#   4654#   4814#   5002#   5161#
                        5320#   5479#   5644#   5789#   5934#   6246#   6393#   6540#   6686#   6861#   7014#   7152#   7291#
                        7427#   7446#   7495#   7611#   8558    8665#   8754#   8827#   9782#   9783   12456
$TMP1   001176          1353#   3327#   3328    3329    3332    3341#   3342    3343    3351#   3352    3353    3363#   3365
                        3369    3370#   3371#   3373    3379#   3381    3385    3386#   3387#   3388#   3389#   3390#   3391#
                        3392#   3394    3400#   3402    3407#   3408#   3409#   3410#   3411#   3412    3420#   3421    3422
                        3428#   3429    3430    3432    3434    3438    3557#   3565    3607#   3633#   3801#   3969#   4139#
                        4299#   4487#   4655#   4815#   5003#   5162#   5321#   5480#   5645#   5790#   5935#   6247#   6394#
                        6541#   6687#   6862#   7015#   7153#   7292#   7428#   7447#   7496#   7612#   8559    8755#   8756
                        8828#   8829    9781#   9788   11479#  11480#  11492   11494   12456
$TMP2   001200          1354#   3608#   7429#   7448#   7613#  11482#  11483#  11506   11509   12456
$TMP3   001202          1355#
$TMP4   001204          1356#
$TN   = 000036           819#   3493    3504#   3534    3544#   3578    3588#   3613    3622#   3781    3790#   3949    3958#
                        4119    4128#   4279    4288#   4467    4476#   4635    4644#   4795    4804#   4983    4992#   5142
                        5151#   5301    5310#   5460    5469#   5625    5634#   5770    5779#   5915    5924#   6060    6069#
                        6145    6154#   6227    6236#   6374    6383#   6521    6530#   6667    6676#   6842    6851#   6995
                        7004#   7133    7142#   7271    7280#   7475    7484#  11888   11928
$TPB    001166         1347#  11841#  11852
$TPFLG  001173         1351#  11790   11852
$TPS    001164         1346#  11839   11852
$TRAP   062202          3247   12313#
$TRAP2  062242         12328#  12339
$TRP  = 000016         12332#  12341#  12342#  12343#  12344#  12345#  12346#  12347   12348#  12349   12350#  12351#  12352#
                       12353#  12354#  12355#
$TRPAD  062254         12322   12339#
$TSTM   001104          1310#
$TSTNM  001116          1324#   7660#  11857   11890#  11917#  11918   11923   11927   11975   12011
$TTYIN  062026         12209   12210   12222   12240   12254   12258#
$TYPBN  057022         11614#  12345
```

```
$TYPOS  057076          11641#  12344
$TYPE   057550          11790#  12332  12340  12429
$TYPEC  057762          11820   11827  11834  11839#  11840  12157
$TYPEX  060030          11845   11847  11850#
$TYPOC  057346          11726#  12341
$TYPON  057362          11725   11728#  12343
$TYPOS  057322          11721#  12342
$UNIT   001234           1374#   3491*   7721   8978
$UNITM  001110           1312#
$USWR   001246           1381#
$VECT1  001272           1406#   3370   3385*   3386   3412*
$VECT2  001274           1407#
$XTSTR  060050          11872#
$$GET4= 000000           7688#
$$SWOB= 000036          11927#  11928  11929#  11930#  11931#  11932#  11933#  11934#  11935#  11936#  11937#  11938#  11939#
                        11940#  11941#  11942#  11943#  11944#  11945#  11946#  11947#  11948#  11949#  11950#  11951#  11952#
                        11953#  11954#  11955#  11956#  11957#
$OFILL  057545          11722*  11726*  11736   11771#
$40CAT= ****** U        11869   11985
.       = 105321         1265#   1269#   1278   1279#  1281#  1283#  1288#  1291#  1297   1298#  1300#  1302#  1314
                         1321#   1363   1481#  1497#  1502#  3240   3255   3256   7673#  7696   7697#  11559#  11695#
                        11852   11926  11927   12011  12014  12018#  12019  12020#  12258#  12259  12266#  12371  12393
                        12450#  12456#
.$ASTA= ****** U        12402   12405
.$X   = 001100           1297#   1302
```

```
CALCLR    808#  3546  3590
CALGET    808#  3667  3699  3742  3835  3867  3910  4003  4036  4080  4173  4205  4247  4333  4365
         4410  4450  4521  4553  4596  4689  4721  4763  4849  4881  4926  4965  5037  5069  5111
         5196  5228  5270  5355  5387  5429  5516  5549  5591  5680  5723  5825  5868  5970  6013
         6113  6195  6282  6336  6429  6483  6576  6630  6734  6796  6910  6964  7050  7094  7188
         7232  7336  7379  7540  7583
CALPUT    808#  3654  3690  3733  3822  3858  3901  3990  4027  4071  4160  4196  4238  4320  4356
         4401  4508  4544  4587  4676  4712  4754  4836  4872  4917  5024  5060  5102  5183  5219
         5261  5342  5378  5420  5503  5540  5582  5668  5714  5813  5859  5958  6004  6102  6183
         6270  6324  6417  6471  6564  6618  6708  6722  6771  6787  6879  6898  6952  7038  7085
         7176  7223  7324  7370  7528  7574
CALSUB    808#  3638  3675  3707  3714  3721  3750  3757  3764  3773  3806  3843  3875  3882  3889
         3918  3925  3932  3941  3974  4011  4044  4051  4058  4088  4095  4102  4111  4144  4181
         4213  4220  4227  4255  4262  4269  4304  4341  4373  4380  4387  4418  4429  4492  4529
         4561  4568  4575  4604  4611  4618  4627  4660  4697  4729  4736  4743  4771  4778  4785
         4820  4857  4889  4896  4903  4934  4945  5008  5045  5077  5084  5091  5119  5126  5133
         5167  5204  5236  5243  5250  5278  5285  5292  5326  5363  5395  5402  5409  5437  5444
         5451  5485  5524  5557  5564  5571  5599  5606  5613  5650  5688  5695  5702  5731  5738
         5745  5754  5795  5833  5840  5847  5876  5883  5890  5899  5940  5978  5985  5992  6021
         6028  6035  6044  6073  6121  6128  6135  6157  6203  6210  6217  6252  6290  6297  6304
         6313  6344  6351  6358  6399  6437  6444  6451  6460  6491  6498  6505  6546  6584  6591
         6598  6607  6638  6645  6652  6692  6742  6749  6756  6804  6811  6818  6827  6867  6918
         6932  6941  6972  6986  7020  7058  7065  7072  7102  7113  7124  7158  7196  7203  7210
         7240  7251  7262  7297  7344  7351  7358  7387  7402  7417  7437  7455  7501  7548  7555
         7562  7591  7602  7620
CNACMS   3493#  3495
COMMEN    950#
DVAVMS   3534#  3536
DVTYMS   3578#  3580
ENDCOM    950#
ERROR     844#  3470  3527  3549  3563  3567  3573  3593  3602  3609  3642  3657  3670  3678  3693
         3702  3710  3717  3724  3736  3745  3753  3760  3767  3778  3810  3825  3838  3846  3861
         3870  3878  3885  3892  3904  3913  3921  3928  3935  3946  3978  3993  4006  4014  4030
         4039  4047  4054  4061  4074  4083  4091  4098  4105  4116  4148  4163  4176  4184  4199
         4208  4216  4223  4230  4241  4250  4258  4265  4272  4308  4323  4336  4344  4359  4368
         4376  4383  4390  4404  4413  4421  4432  4438  4453  4460  4464  4496  4511  4524  4532
         4547  4556  4564  4571  4578  4590  4599  4607  4614  4621  4632  4664  4679  4692  4700
         4715  4724  4732  4739  4746  4757  4766  4774  4781  4788  4824  4839  4852  4860  4875
         4884  4892  4899  4906  4920  4929  4937  4948  4954  4968  4975  4980  5012  5027  5040
         5048  5063  5072  5080  5087  5094  5105  5114  5122  5129  5136  5171  5186  5199  5207
         5222  5231  5239  5246  5253  5264  5273  5281  5288  5295  5330  5345  5358  5366  5381
         5390  5398  5405  5412  5423  5432  5440  5447  5454  5489  5506  5519  5527  5543  5552
         5560  5567  5574  5585  5594  5602  5609  5616  5654  5671  5683  5691  5698  5705  5717
         5726  5734  5741  5748  5759  5799  5816  5828  5836  5843  5850  5862  5871  5879  5886
         5893  5904  5944  5961  5973  5981  5988  5995  6007  6016  6024  6031  6038  6049  6077
         6105  6116  6124  6131  6138  6161  6186  6198  6206  6213  6220  6256  6273  6285  6293
         6300  6307  6317  6327  6339  6347  6354  6361  6403  6420  6432  6440  6447  6454  6464
         6474  6486  6494  6501  6508  6550  6567  6579  6587  6594  6601  6611  6621  6633  6641
         6648  6655  6696  6711  6725  6737  6745  6752  6759  6774  6790  6799  6807  6814  6821
         6832  6871  6882  6901  6913  6921  6930  6935  6945  6955  6967  6975  6984  6989  7024
         7041  7053  7061  7068  7075  7088  7097  7105  7116  7122  7127  7162  7179  7191  7199
         7206  7213  7226  7235  7243  7254  7260  7265  7301  7327  7339  7347  7354  7361  7373
         7382  7390  7405  7411  7420  7440  7449  7458  7505  7531  7543  7551  7558  7565  7577
         7586  7594  7605  7614  7623
ESCAPE    950#
GETCS1    808#
```

DZRMDA.P11 29-JUL-77 14:07 CROSS REFERENCE TABLE -- MACRO NAMES

```
GETDB      808#
GETDS      808#
GETDT      808#   3596
GETMR1     808#
GETPR1     950#
GETSWR     808#    950#   3285#   3474
MULT       950#
NEWTST     950#   3493   3534   3578   3613   3781   3949   4119   4279   4467   4635   4795   4983   5142   5301
          5460   5625   5770   5915   6060   6145   6227   6374   6521   6667   6842   6995   7133   7271   7475
NWTST      808#   3496   3538   3582   3616   3784   3952   4122   4282   4470   4638   4798   4986   5145   5304
          5463   5628   5773   5918   6063   6148   6230   6377   6524   6670   6845   6998   7136   7274   7478
POP        950#   3520   3521   3525   3526   3559   3560   3571   3572   8119   8338   8339   8352   8495   8496
          8497   8569   8571   8572   8573   8574   8675   8677   8678   8679   8708   8709   8710   8783   8785
          8786   8787   8788   8789   8790   8850   8852   8853   8854   8855   8856   8857   8879   8881   8921
          8923   8924   8925   8926   9074   9078   9222   9806   9807   9808   9809  10069  10071  10072  10073
         10133  10695  10696  11598  11682  12300  12380  12381  12444  12445
PUSH       950#   3506   3507   3551   3553   8096   8321   8322   8345   8363   8364   8365   8521   8523   8524
          8525   8526   8597   8598   8599   8600   8692   8694   8695   8742   8743   8744   8745   8746   8747
          8748   8815   8816   8817   8818   8819   8820   8821   8863   8864   8892   8893   8894   8895   8896
          9070   9219   9770   9771   9772   9773  10053  10055  10056  10057  10127  10688  10689  11578  11641
         12279  12361  12367  12405  12407  12428
PUTCS1     808#
PUTDC      808#
PUTER1     808#
PUTMR1     808#
REPORT     950#
RGBFMC     808#   1427   1454
SCOPE      845#   3539   3583   3617   3785   3953   4123   4283   4471   4639   4799   4987   5146   5305   5464
          5629   5774   5919   6064   6149   6231   6378   6525   6671   6846   6999   7137   7275   7479
SETPRI     950#   3230   3487  11534  11540  12187
SETTRA   12332#  12341  12342  12343  12344  12345  12347  12349  12350  12351  12352  12353  12354
SETUP      950#   3235
SKIP       950#
SLASH      950#
SPACE      950#
STARS      950#   1276   1294   1296   1303   1317   1363   1366   3493   3495   3534   3537   3578   3581   3613
          3615   3781   3783   3949   3951   4119   4121   4279   4281   4467   4469   4635   4637   4795   4797
          4983   4985   5142   5144   5301   5303   5460   5462   5625   5627   5770   5772   5915   5917   6060
          6062   6145   6147   6227   6229   6374   6376   6521   6523   6667   6669   6842   6844   6995   6997
          7133   7135   7271   7273   7475   7477   7651   7700   7901   7914   7926   7948   7977   7999   8027
          8091   8357   9117   9125   9210   9593  11562  11607  11631  11698  11775  11854  11959  12013  12090
         12105  12176  12200  12269  12307  12357  12373  12400
SWRSU      950#   3257#
TAGS       808#   1422
TRMTRP   12332#
TYPBIN     950#
TYPDEC     950#   7674   7681
TYPNAM     808#    950#   3278
TYPNUM     950#
TYPOCS     950#   3373   3394   7721   7729   7738   7744
TYPOCT     950#   3359  12118
TYPTXT     950#   7670   7677
XPER       808#   1524   1531   1538   1545   1552   1559   1566   1573   1580   1587   1594   1601   1608   1615
          1622   1629   1636   1643   1650   1657   1664   1671   1678   1685   1692   1699   1706   1713   1720
          1727   1734   1741   1748   1755   1762   1769   1776   1783   1790   1797   1804   1811   1819   1826
          1833   1840   1847   1855   1863   1871   1878   1885   1892   1899   1906   1913   1920   1927   1935
```

```
                1942    1949    1956    1963    1970    1977    1984    1991    1998    2005    2012    2019    2026    2033    2040
                2047    2054    2061    2068    2075    2082    2089    2096    2103    2110    2117    2124    2131    2138    2145
                2152    2159    2166    2173    2180    2187    2194    2201    2208    2215    2222    2229    2236    2243    2250
                2257    2264    2271    2278    2285    2292    2299    2306    2313    2320    2328    2335    2342    2349    2356
                2363    2370    2377    2384    2391    2398    2405    2412    2419    2426    2433    2440    2447    2454    2461
                2468    2475    2482    2489    2496    2503    2510    2517    2524    2531    2538    2545    2552    2559    2566
                2573    2580    2587    2594    2601    2608    2615    2622    2629    2636    2643    2650    2657    2664    2671
                2678    2685    2692    2699    2706    2713    2720    2727    2734    2741    2748    2755    2762    2769    2776
                2783    2790    2797    2805    2812    2820    2827    2834    2842    2850    2858    2867    2875    2883    2890
                2897    2904    2911    2918    2925    2932    2939    2946    2953    2960    2967    2974    2981    2988    2995
                3002    3009    3016    3023    3030    3037    3044    3051    3058    3065    3072    3079    3086    3093    3100
                3107    3114    3121    3128    3135    3142    3149    3156    3163    3170    3177    3184
SSCMRE          1315#
SSCMTM          1315#   1352    1353    1354    1355    1356
SSESCA           950#
SSNEWT           950#   3493    3534    3578    3613    3781    3949    4119    4279    4467    4635    4795    4983    5142    5301
                5460    5625    5770    5915    6060    6145    6227    6374    6521    6667    6842    6995    7133    7271    7475
SSSET          12332#  12341   12342   12343   12344   12345   12347   12349   12350   12351   12352   12353   12354
SSSETM          3273#
SSSKIP           950#
.EQUAT           808#    840
.HEADE           808#    809
.SETUP           808#   1262
.SWRHI           808#    820
.SWRLO           808#    832#
.$ACTI           808#   1274
.$APTB           808#   1364#
.$APTH           808#   1292
.$APTY           808#  12398
.$CATC           808#   1263
.$CMTA           808#   1315
.$DIV            808#
.$EOP            808#   7649
.$ERRO           808#  11957
.$ERRT           808#
.$MULT           808#
.$POWE           808#  12355
.$RAND           808#
.$RDDE           808#
.$RDOC           808#  12267
.$READ           808#  12011
.$SAVE           808#  11560
.$SCOP           808#  11852
.$SIZE           808#
.$TRAP           808#  12305
.$TYPB           808#  11605
.$TYPD           808#  11629
.$TYPE           808#  11773
.$TYPO           808#  11696

. ABS.   105321     000

 ERRORS DETECTED:  0
```

```
 DSKZ:DZRMDA.BIN,DSKZ:DZRMDA.SEQ/DOC/SOL/CRF=DSKM:DZRMDA.P11
 RUN-TIME: 64 60 4 SECONDS
 RUN-TIME RATIO: 1299/129=10.0
 CORE USED:  33K  (65 PAGES)

 DOCUMENT PAGES:  311


EOF1DZRMDASEQ              00010000      770804            PDP10 411
```